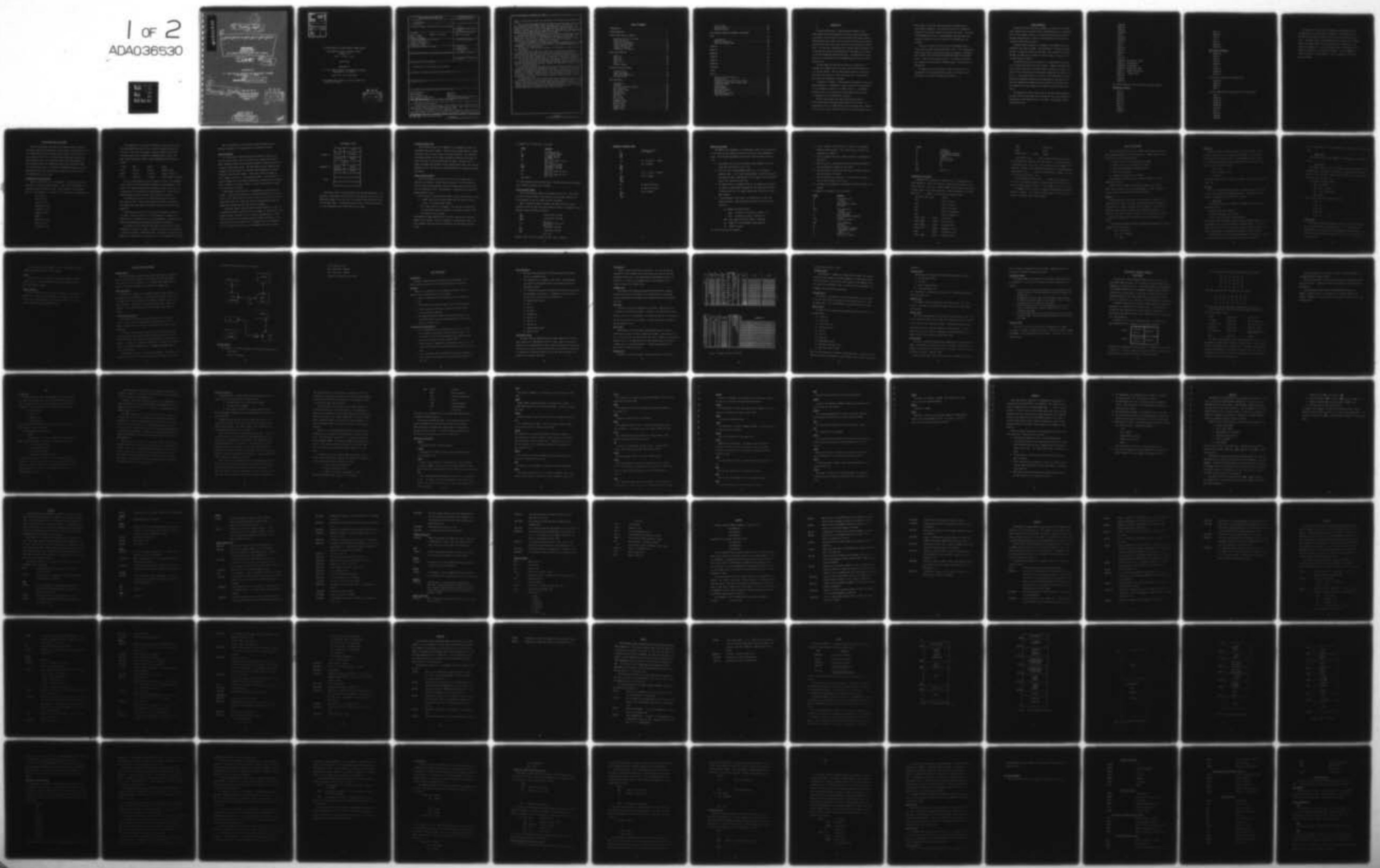
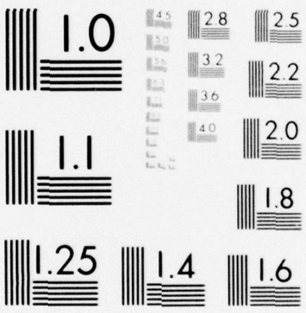
AD-A036 530 CLEMSON UNIV S C COLL OF ENGINEERING F/G 13/2
A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN. APPENDIX VII--ETC(U)
AUG 75 T L DRAKE DADA17-72-C-2152

UNCLASSIFIED NL

1 OF 2
ADA036530





(12)

(9) Final rept.

(6)

A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN.

~~FINAL REPORT~~
APPENDIX VIII.
DOCUMENTATION
DATA HANDLING SYSTEM PROGRAMMER'S MANUAL.

(11) AUGUST 1975 (12) 118p.

SUPPORTED BY

U.S. ARMY MEDICAL RESEARCH AND DEVELOPMENT COMMAND
WASHINGTON, D.C. 20314

(15) ~~CONTRACT NO.~~ DA DA 17-72-C-2152

(10) Thomas L. Drake

PUBLICATION OF
ENGINEERING RESEARCH
CLEMSON UNIVERSITY
CLEMSON, S.C.

408 039

A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN

Data Handling System Programmer's Manual

APPENDIX VIII

Author: Thomas L. Drake

August 1975

Supported by

U. S. Army Medical Research and Development Command
Washington, D. C. 20314

Contract No. DA 17-72-C-2152

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Final Report	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Systems Analysis of Water Quality Survey Design		5. TYPE OF REPORT & PERIOD COVERED Final
		6. PERFORMING ORG. REPORT NUMBER Final
7. AUTHOR(s) Lyle C. Wilcox Thomas L. Drake Ralph W. Gilchrist Bobby E. Gilliland		8. CONTRACT OR GRANT NUMBER(s) DADA 17-72-C-2152
9. PERFORMING ORGANIZATION NAME AND ADDRESS Clemson University College of Engineering Clemson, SC 29631		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE August 1975
		13. NUMBER OF PAGES 862
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Systems Analysis
Water Quality
Water Quality Survey
Water Quality Survey Design
Army Ammunition Plants
Modeling
Simulation
Ammunition Manufacturing Processes
Water Quality Measurements

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This is the final report of a three year project titled, "A Systems Analysis of Water Quality Survey Design."

In this project a study was made of water quality surveys conducted by the United States Army Environmental Hygiene Agency (AEHA). Mainly data and reports from studies of Army Ammunition Plants (AAP) were used.

The focus of this project was the development of computer aided procedures which would assure efficient use of manpower and equipment and assure that the measurements taken give a reasonalbe representation of the system. Planning the

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unlimited
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

survey, conducting the survey and reporting on the survey were included in the study.

The site modeling program models the manufacturing processes which contribute pollutants to the system, models the sewer system, and models the treatment system including acid or caustic neutralization, settling ponds, and domestic treatment. The inputs to the model are the production levels of the manufacturing processes and the outputs are the predicted pollutant measurement values at each possible measure point in the system.

The resource matching program accepts data defining proposed measurements and matches these against the available time, manpower, and equipment. The output lists the pollutant to be measured at each measure point, the total commitment of time for each analyst and for each piece of equipment. Note is made of any overcommitment of manpower or equipment.

The model refinement or updating program accepts measurements taken during a preliminary survey or during a regular survey and computes suggested new parameters for the process models.

The indicator model program evaluates the performance of sanitary treatment facilities.

The program uses design data, data from the operating log and/or data generated during the survey and computes key operational characteristics. Comparing these with desirable values as cited in design books and manuals will give the survey planner insight into the operation of the system and suggest the need for more survey measurements or the need for changes in operation.

A system was developed for automatic instrumentation of pH, conductivity, and other parameters which use strip charp recordings. Interface hardware was selected and purchased and interface software was developed for direct connection to a digital computer.

A data handling system was developed for use during and after the survey. A PDP8-OS/8 and peripheral equipment was purchased. Software was developed to perform data handling functions and to direct the user in application of the software. The program accepts raw data from the analytical chemist and performs data conversions, transcriptions, and data logging functions. Output is available in several forms as may be needed for various reports during and at the end of the survey.

Recommendations are: the survey planner should obtain sufficient data in a proliminary survey to model and analyze the site; measurements should be automated to the maximum extent possible; data handling should be delegated to the computer when the operations are well defined and repetitive. The programs, software and hardware included here will assist the survey planner in following these recommendations and design a more effective survey.

TABLE OF CONTENTS

INTRODUCTION

A Data Handling System is described which handles on-site water quality survey data measured by the Army. Data and commands are entered into the system on-line via either the Teletype, Hewlett-Packard optical mark reader, Yellow Springs S-C-T meter, Talos graphic digitizer, DEC manual data entry station, or Fisher Scientific pH/ion meter. The digitizer, manual data entry station, specific conductivity instrumentation, and pH instrumentation are interfaced to the system via the DEC PDM-7Ø programmable data mover. The Data Handling System may be at a fixed location with a telephone data link and /or may be found on-site.

The Data Handling System may be viewed as a collection of programs which operate within the supervision of the OS/8 executive on a 16K DEC PDP-8/E. Each of these programs operate in conjunction with data files found on the mass storage devices. These programs are written using OS/8 Basic and PAL8.

A sequence of programs is often executed to complete a given data handling operation. OS/8 BATCH is used to automate the specification of this sequence of operation via a BATCH input file. OS/8 BATCH provides users of the system with a batch processing monitor that is integrated into the OS/8 Monitor structure.

Several system tables are used by the system software to define a particular data handling application to the system. The system definition file (SYSDEF.AR) provides information which includes lists of valid ID tags, upper and lower limits for measurement values, cont

data formats, and various heading and column information used during report generation. The translation table file (TRNTBL.AR) contains the translation tables for the mark sense cards. The method file (METHOD.AR) specifies for each parameter the particular data conversion algorithm within the function file (FNCTN.AR) to use on input.

The data files and information files are OS/8 BASIC numeric files while all other files are OS/8 ASCII files. The ID information within these numeric files are 6-bit positive integers. OS/8 BASIC user functions are provided to pack and unpack these 6-bit integers within a 36-bit floating point variable. In addition, OS/8 BASIC user functions are provided for formatted output, interization, and PDM-7Ø input.

The chapters within this manual include a discussion of the implementation of the data handling programs, file formats, mark card design, and industry compatible type formats.

SYSTEM GENERATION

The Data Handling System may be viewed as a collection of programs which operate within the supervision of the OS/8 executive on the PDP-8/E. Each of these programs operate in conjunction with data files found on the mass storage devices. Several of the data handling functions are provided by OS/8 System Programs.

The data handling software is provided on four DECtapes which are marked SYS, DSK, SRC, and INSRC. The OS/8 Version III executive system software is supplied on additional DECtapes. The programs on the tapes SYS,DSK, and INSRC can be created from the SRC tape in conjunction with the OS/8 executive software.

The OS/8 system monitor has been built to make DECtape unit Ø and unit 1 respectively the devices SYS: and DSK:. Likewise, the data handling functions, described in the Data Handling System User's Manual, expects to find the tapes SYS and DSK mounted respectively on DECtape units Ø and 1. The program INPUT.SV on SYS and the file SYSDEF.AR are highly application dependent and hence must be generated for each application. The files METHOD.AR and TRNTBL.AR may also change for each application. The remaining files and programs should remain the same for each application.

The DECtape SRC contains the sources of the data handling software. The tapes SYS, DSK, and INSRC can be created from this source DECtape in conjunction with the OS/8 Version III software. This DECtape contains the following files:

REPORT.BA	
MERGE.BA	
IDGEN.BA	
DAHDLR.BA	
TDS.BA	
TDSINF.BA	
TDSFNV.BA	
SORT.PA	
CRH.PA	
GETPUT.PA	
MERGE.BI	
REPORT.BI	
TDS.BI	
EDITS.BA	
SYSDEF.AR	(Letterkenny Survey)
TRNTBL.AR	(Letterkenny)
METHOD.AR	(Letterkenny)
LETDAT.AA	(Letterkenny Data)
BADSD.AR	(Badger System Def.)
BADAT.AA	(Badger Data)
FNCTN.AR	
WTHDL.BA	
GENCRH.BI	

The DECtape marked SYS must contain the following programs:

<u>OS/8 System Programs:</u>

CCL.SV

BATCH.SV

BASIC.SF

BASIC.AF

BASIC.FF

BASIC.VF

BRTS.SV

BCOMP.SV

BLOAD.SV

BASIC.SV

DIRECT.SV

FOTP.SV

PIP.SV

PAL8.SV

ABSLDR.SV

NONEAE.BN

Data Handling Programs:

REPORT.SV

MERGE.SV

INPUT.SV

SORT.SV

DAHDLR.SV

TDS.SV

TDSINF.SV

TDSFNV.SV

WTHDL.SV

The tape DSK contains the following files:

SYSDEF.AR

REPORT.BI

MERGE.BI

TDS.BI

The tape INSRC contains the following files and programs:

CRH.PA

GETPUT.PA

SYSDEF.AR

METHOD.AR

TRNTBL.AR

FNCTN.AR

EDIT.SV

EDITS.SV

IDGEN.BA

GENCRH.BI

The OS/8 ASCII files SYSDEF.AR, METHODS.AR, and TRNTBL.AR define the application to the data handling software. The programs EDITS.SV and EDIT.SV are present on INSRC for editing the three system definition files. Once these files have been defined and are present on the DECtape INSRC, the program INPUT.SV can be created for this application. The generation procedure for this program assumes that the DECtapes SYS and INSRC are respectively on DECtape units Ø and 1. The file GENCRH.BI defines the steps to create INPUT.SV. The file SYSDEF.AR on DSK must also be updated.

SYSTEM TABLES AND FILE FORMATS

Several system tables are used by the system software to define a particular data handling application to the system. The system definition file (SYSDEF.AR) provides information which includes lists of valid ID tags, upper and lower limits for measurement values, data formats, and various heading and column information used during report generation. The translation table file (TRNTBL.AR) contains the translation tables for the mark sense cards. The method file (METHOD.AR) specifies for each parameter the particular data conversion algorithm within the function file (FNCTN.AR) to use on input.

System Definition File Format:

The system definition file, DSK:SYSDEF.AR, is required to define the application to the data handling system. This file is an OS/8 ASCII file containing several character strings. A Null character string segments this file into several lists. These lists, in the order of appearance, are

Day ID Lists

Point ID Lists

Period ID Lists

Parameter ID Lists

Discrete ID Lists

Comment ID Lists

Header

Chemists ID Lists

Quality ID Lists

Accuracy ID Lists

The Parameter ID list contains 6 character strings per list entry while the remaining primary ID lists, contain 3 character strings per list entry. The first 3 character strings per primary ID list entry provide alternate ways of externally representing this ID tag. The name, length, and usage of each of these representations are shown as follows:

Order	Name	Length	Usage
First	Short ID	1-6 char.	Normally used
Second	Long ID	1-24 char.	Header of Reports
Third	Column ID	1-6 char.	Column header in Reports

The 1st three character strings for each list entry in the Parameter ID list defines a minimum, maximum, and format for each parameter. The format is an integer and specifies the number of places to the right of the decimal point to be used during report generation. The minimum and maximum defines the lower and upper limits during a print-plot.

The header must be 3 lines in length even though one or more of these lines are null. Each line must contain less than 72 characters. This header labels each page of the report and print-plot during report generation.

The Accuracy and Quality ID lists contain 2 character strings per list entry. The first string is a 1 to 6 characters in length and is normally used. The second string is 1 to 24 characters in length and provides an alternate way of representing the same ID information.

The Chemist ID list identifies the name of the chemist and contains 2 character strings per list entry. The first string is 1 to 6 characters in length and provides an abbreviated name. The second string permits the chemist to be identified with a name which is 1 to 24 characters in length.

The file SYSDEF.AR can be edited with either the BASIC program EDITS.BA, designed for this application, or the OS/8 Text Editor.

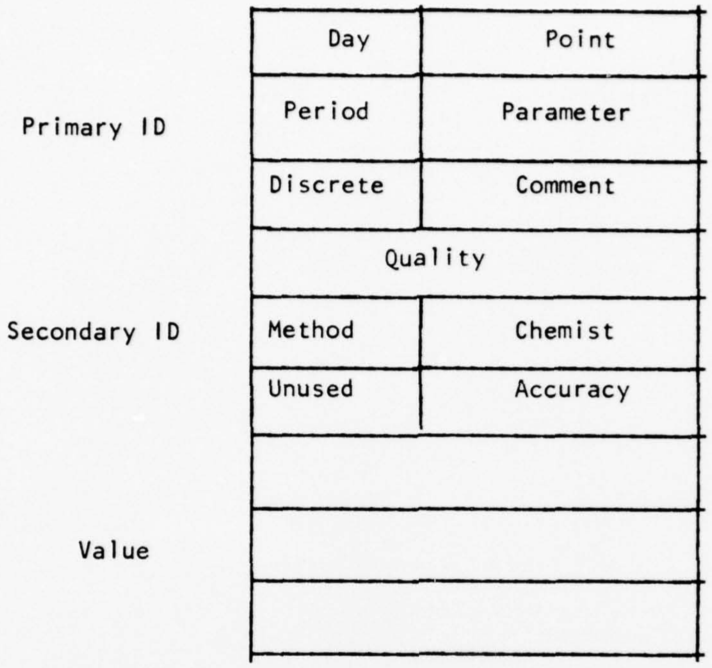
Data File Formats:

All files under the OS/8 executive are allocated in multiples of 256 12-bit word blocks. Data in a numeric file, compatible with OS/8 BASIC, places 85 three-word floating-point numbers in each block with the last word in each block unused. The data handling system stores 28 measurements within each block and represents each measurement with three successive floating-point numbers. The primary ID tags are packed as 6-bit integers in the first floating-point variable. The secondary ID tags are packed within the second floating-point variable while the numeric value is placed in the last variable. The last floating-point variable within each block is unused.

Each primary ID tag internally is represented as a 6-bit positive integer. A zero value indicates that no tag was entered for this particular ID. This integer points to an entry within a valid ID list within the system definition file. BASIC user functions are provided to pack and unpack these ID tags from the floating-point variables

The Method, Chemist, Accuracy, and the unused secondary ID tags are represented internally with 6-bit positive integers in the same way as the primary ID tags. The Quality ID is internally represented by a single 12-bit word with one bit assigned to each ID within the Quality ID list. The left most bit of the Quality ID is assigned to the first element in this list while the right most bit is assigned to the 12th list entry.

MEASUREMENT FORMAT



Some blocks within a file may contain less than 28 measurements. The absence of a measurement within a block will be noted by a zero primary ID. Measurements appear within the file in an arbitrary order unless the data file has been sorted. A sequential search of this file must be performed to find a measurement with a specified set of ID tags.

Translation Table File:

The translation table file, TRNTBL.AR, is an OS/8 ASCII format file and contains two sets of tables. The first set of tables defines a unique translation table for each column of the mark sense card. The translated card image provided via this column translation table can be divided into fields with each field being delimited by a space. The second set of tables translates the character strings, defining each field, into a second character string. The data entry software processes this second translated card image.

Column Translate Tables:

The first character string within this file is an integer which identifies the number of columns found on the card. The file contains a table for each of these columns with each of these tables being delimited by a null character string. Each character is translated into a variable length character string.

The first entry in each translate table is one of the following 2 cases.

1. Integer specifying the column number with the labeling starting at zero. A translate table follows.
2. N = P where N and P are integers. This notation indicates that the translation table for column N is the same as column P (P<N) and hence will not be given.

The character strings which follow in the first case are the translation tables for this column. The first character in a character string is the input character while the remaining characters are the output character string.

An example of a 5 column card is as shown:

Table	Comments
5	5 columns on card
0	Col. 0 table follows
1AB	1 changed into AB
2BC	2 changed into BC
	Null line
1=0	Use col. 0 table for col. 1
	Null line
2	Col. 2 table follows
Z123	Translate Z into 123
Y456	Translate Y into 456
	Null line
3=2	Use col. 2 table for col. 2
	Null line
4=0	Use col. 0 table for col. 4
	Null line
Field Tables	

All input characters which are not found in column translate are considered to be illegal by the data entry software.

Field Translate Tables:

Each field translate table is delimited by a null line. The absence of a field translate table is the character string @@ and indicates the field character string is already correctly translated.

When a translation table is present, the table provides an output character string for each valid input character string. An example of field translate tables are shown as follows:

AB	Translate AB into 1234
1234	
CD	Translate CD into 2345
2345	
	Null line
@@	No table for this field
	Null line
A	Translate A into TLD
TLD	
B	Translate B into ABC
ABC	
	Null line

Usually, there is one field table for each field on the card.

Example of Translate Table

4	4 Columns on Card
0	Col. 1 Table
1AB	
2BC	
3CD	
1=0	Col. 2, Use Col. 1 Tables
2	Col. 3 Tables
ACD	
BDE	
EFG	
3=2	Col. 4, Use Col. 3 Tables
AB	Field 1 Tables
CDEF	
BC	
EFGH	
@@	No Table for Field 2
@@	No Table for Field 3
CD	Field 4 Tables
1234	
DE	
6ACD	

Methods File Format:

The methods file, METHOD.AR, is an OS/8 ASCII format file and identifies to the system the data reduction algoritym to use for a given parameter on input. The following paragraphs give a set of rules for constructing this file.

1. The methods file contains one block for each parameter with each block being delimited by a null line.
2. The first line of each parameter block is a 1 to 6 character parameter name. If this particular parameter is used during survey, the parameter name in the Methods file must agree with the parameter name in the system definition file.
3. The second entry in a parameter block is an integer which specifies the number of data reduction methods for this particular parameter. A value of zero for this integer denotes that the end-of-file has been reached.
4. In each parameter block, there is a method block for each data reduction method. Each method block contains at least 4 entries which are

N_c = Number of Constants where

$N_c \geq 0$	Constants are not given in methods file
$N_c < 0$	Constants are given in methods file

FNR = Data reduction function number in FNCTN.AR

N_m = Number of raw measurement values expected

N_t = Number of tables

All of these entries are integers.

5. If N_c is negative, then the next N_c entries in the methods block are these constants. For a positive N_c, no entries are allocated for constants.

6. If N_t is greater than zero, a block of entries is allocated for each table.

7. The first entry in each table block gives the maximum number of entries available in memory for this table.

8. The second entry in each table block gives the actual number of entries present in the table.

9. The next entries are the actual x, y entries for the table. A line is used for each x and y entry.

10. The table is terminated by the parameter with the name END with 0 methods.

An example for a methods file is as follows:

Table	Comments
PH	PH parameter block
1	1 method
0	no constants
1	Data reduction formula 1
0	No tables
TALK	Total Alkalinity
1	1 method
-1	1 constant given
2	Data reduction
2	2 raw measurements expected
0	No table
.307	Normality constant
P04	Total Phosphates
1	1 method
0	No constants
4	Data reduction formula 4
1	1 measurement expected
1	1 table
4	Table size is 4
0	No entries in table

```
PHENOL
1                              1 method
0                              No constants
5                              Data Reduction formula5
2                              2 measurements expected
1                              1 table
6                              Table size is 6
2                              2 entries given
0                              X1
0                              X1
30                             X2
40                             Y2

END                            End of file
0
```

Information File Format:

A temporary file, called the information file, is generated during report generation. This file usually is created on device INFO: with the name of INFO.IN. This file is a BASIC numeric file and contains one record for each report or print-plot specified. The 85 floating-point variables, D(0), D(1), . . ., D(84), in this record are coded as follows:

D(0), D(1), D(2), D(3)	Unused
D(4)	Order of Day ID
D(5)	Order of Point ID
D(6)	Order of Period ID
D(7)	Order of Parameter ID
D(9)	Order of Comment ID
D(10), D(11), ...,D(20)	Day ID Lists
D(21), D(22), ...,D(30)	Point ID Lists
D(32), D(33), ...,D(42)	Period ID Lists
D(43), D(44) . . . .,D(53)	Parameter ID Lists
D(54), [illegible], ...,D(64)	Discrete ID Lists
D(65), D(66), ...,D(75)	Comment ID Lists

D(76)	Segment Size
D(77)	Option
D(78), D(79), ... ,D(84)	Unused

The variables, D(4), . . ., D(9), determine the order of the sort during the SORT step and hence the report format. For continuous data, the variables, D(4), D(5), D(6), D(7) are unique positive integers with values of 1 through 4, while D(8) and D(9) are zero. For discrete data, the variables, D(4), . . ., D(8), are unique positive integers with values of 1 through 5 while D(9) is zero.

For the variables, D(10), D(11), . . ., D(75), which contains the ID lists, six ID tags are packed into each variable with a zero ID tag used to indicate the end of each list. The Comment ID list is not used.

The segment size is a positive integer which determines the number of columns in a report. For a report, the option has a value of zero while a value of 1 indicates the print-plot option.

BASIC USER FUNCTIONS

The data handling system requires 5 BASIC user functions which have been programmed especially for this application. These functions provide BASIC with the following capabilities:

1. Unpack ID tags from a floating point variable.
2. Pack ID tags into a floating point variable.
3. Accept and decode PDM-70 character strings.
4. Print using F formats.
5. Integerize a floating point variable.

The programmer must define these functions to BASIC with a UDEF statement as follows:

```
5 UDEF  UFA(I,J), UFB(I,J,K), UFC(I,A$), UFD(I,J,K,L), UFE(I)
```

BASIC uses the order in which a function appears in a UDEF statement and not the function name to determine the user function.

UFA(I,J)

The data handling system packs six 6-bit positive integers within a floating point variable. This user function provides the programmer with the capability to unpack these 6-bit positive integers. The variable I determines which integer to unpack from the floating variable J. The value of I must then be either 0,1,2,3,4, or 5. The value of the function is a positive integer with a minimum value of zero and a maximum value of 63.

The following example shows BASIC statements which place the 6 integers packed in A into the array I1.

```
100   FOR I=0 TO 5
110   I1(I) = UFA(I,A)
120   NEXT I
```

UFB(I,J,K)

The user function L = UFB(I,J,K) packs the 6-bit positive integer J as the Ith integer into the floating point variable L. The remaining five 6-bit positive integers found in L are obtained from the variable L. The following example packs the integers I1(0), I1(1), I1(2), I1(3), I1(4), and I1(5) into the variable A.

```
        200  FOR I=0
        210  A = UFB(I,I1(I),A)
        220  NEXT I
```

The minimum value of each of these 6-bit integers is zero while the maximum value is 63.

UFC(I,A$)

The user function UFC(I,A$) was coded to receive input from the Talos digitizer via the PDM-70 Programmable Data Moves. The operation of this function is dependent on the value of the variable I.

Case: I=0

The Teletype handler used by BASIC is replaced with a special Teletype handler which:

1) doesn't echo input characters
2) ignores characters such as rubout
3) has a time-out feature to terminate input

The time-out feature terminates input by placing a RETURN character in the input buffer after a prescribed amount of time has elasped. This time is approximately 1 second after one or more character has been received. This time is approximately 1 minute before a character has been received.

The PDM-70 doesn't provide the RETURN character to terminate a character string when transferring data from the 8 digit BCD input cards.

This revised handler thus uses the absence of a character string to terminate input.

Case: I = 1

The Teletype handler used by BASIC is restored back to its original version.

Case: I $\geq$ 2

The numeric value of the I-1st character string A$ is returned as the function value. The value of the function is always a positive integer equal to or less than 63. A value of zero indicates that I points beyond the end of the character string.

For example, the BASIC program

```
300   A$ = "AB12"
310   FOR I = 0 TO 10
320   I1(I) = UFC(I+2,A$)
330   NEXT I
```

would provide the following values for I1(I):

```
I1(0) = 1
I1(1) = 2
I1(2) = 49
I1(3) = 50
I1(I) = 0        I >= 4
```

UFD(I,J,K,L)

This function outputs the variable L to file K using a field width of I and J places after the decimal point. If J is zero, then the decimal point is not output and is not included in the field count. The file K must be open and have a value of 0,1,2,3, or 4. Only the minus sign is printed.

This function rounds the number. For J=1, the numbers 7.3516 and 7.4995 both produce a value of 7.4 on output.

UFE(I)

The operation of this function is exactly the same as the BASIC INT(X) function. The usé of UFE(I) function avoids BASIC overlays in many data handling programs.

General Comments

The file BASIC.UF contains these user functions. The source file is BASUF.PA and is found on the source tape. If required, this source file can be assembled, loaded, and then saved as BASIC.UF. This user function occupies locations 3400-4577 octal.

TELETYPE APPLICATION NOTES

Teletype Parity:

The Teletype keyboard generates even parity which may cause problems when running maintenance programs. The standard DEC Teletype keyboard forces the parity bit to always be "1". The paper tape reader sends the parity bit information punched in the 8-hole of the paper tape.

Paper Tape Reader:

The paper tape reader on the Teletype has been slightly modified to be compatible with the PDP-8/E. A relay has been inserted in series with the reader on/off switch to permit the computer to remotely operate this switch function. The KL8-E with Teletype interface cable supports this function. This relay must be energized for the paper tape reader to operate.

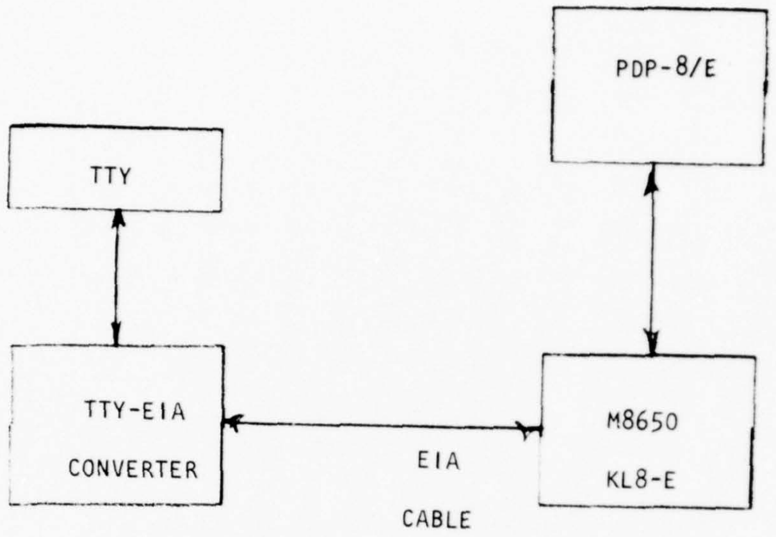
EIA RS-232 Compatibility:

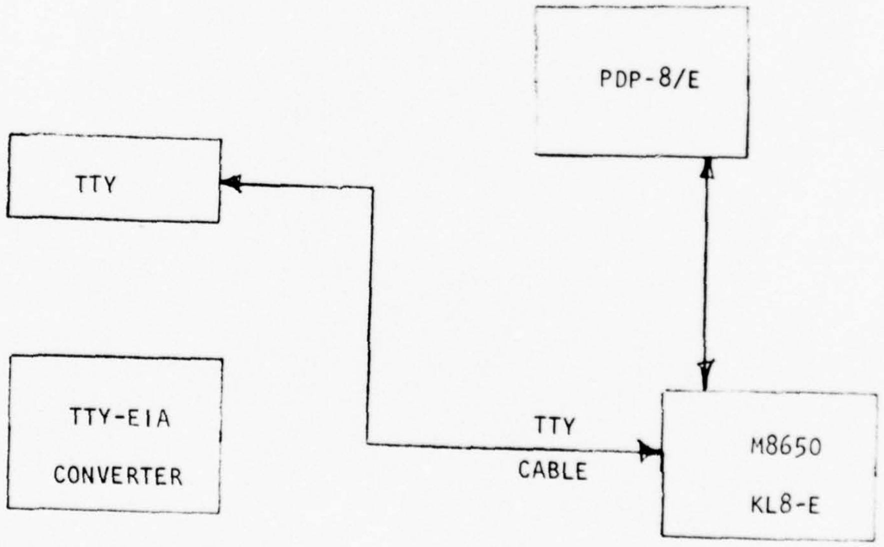
The Teletype is designed to send and receive serial information which is coded as currents having approximate values of 0 and .020 amperes. Modems and other terminals are generally EIA RS-232 (voltage) compatible.

A Teletype - EIA converter is present in the base of the Teletype to provide the operator with the option of converting the Teletype to EIA RS-232 levels. Both the HP card reader and modem are EIA RS-232 compatible. In addition, the computer interface will accept either Teletype (current) or RS-232 (voltage) levels. The paper tape reader does not work in RS-232 mode.

Three cables should exit the base of the Teletype. One cable is the Teletype I/O cable while the remaining two are the converter I/O cables.

The following block diagrams should be useful.





<u>Teletype Options</u>

The teletype is internally strapped to provide the following options:

Full Duplex

Even Parity Keyboard

20 Milliampere Current	
Paper Tape Reader	XON/XOF
Paper Tape Punch	XON/XOF
Automatic Carriage Return-Line Feed	

MARK CARD DESIGN

Introduction:

The steps necessary to design a mark card are presented. This design can be printed by IBM. An actual card design is discussed.

Procedure:

Cards for this application can be acquired from IBM. The procedure for this card acquisition are as follows:

1. Submit a rough sketch of the proposed card to the IBM salesman.
2. IBM transforms the customers sketch into a final card design.
3. Prior to printing, IBM submits a proof of the final design to the customer for approval.
4. Once approval has been obtained, the cards are printed. Approximately 60 days are required from submission of the rough sketch to the final card.

Card Size and Column Spacing:

1. It is recommended that a standard card size (3 1/4 x 7 3/8) and a standard column spacing (40/80 columns) be used. This will permit the cards to be prepunched with conventional punch facilities (keypunch or punch on UNIVAC 1108).
2. The card reader is capable of accepting card sizes up to 11 1/8 inches long with vertical columns positioned by the use of clock marks.
3. It is recommended that a standard 40 column card be adopted. A closer spacing (more columns) yields card marking problems for the user.

Card Information:

1. The data handling software classifies each card as a command card or a measurement card.
2. The command card is a command to the system. The measurement card contains a measurement which is specified by an ID and measurement values.
3. It is desirable to have one card design to serve both the command and measurement card format. To accomplish this, the card must contain fields for the following information:
   a) Command Specification
   b) Day ID
   c) Point ID
   d) Period ID
   e) Parameter ID
   f) Discrete ID
   g) Quality ID
   h) Accuracy ID
   i) Comment Specification
   j) Numeric Field

Card Column Format:

The Hewlett-Packard 7260A Optical Mark Reader expects to find each column coded with a 128-character Hollerith code. Each card character is converted to its equivalent 8-bit ASCII character before being transmitted. The blocks within each column when marked must yield a Hollerith character. If standard 40- or 80-column spacing is not used, then clock marks must present on the bottom of the card.

Past Designs:

Figure 1 shows two different card designs. The first design was designed for the 1973 Badger AAP survey and was also used for the 1974 Letterkenney AAP survey. The second design was designed to correct the problems encountered with the Badger design. The remainder of this chapter discusses this second design.

Command Field:

The program INPUT recognizes approximately 60 different commands. Usually two columns are allocated to the specification of the command while an additional column separates the command field from the adjacent day field.

Day Field:

A 39 day survey can be handled with one column. Punching position 1 through 9 are assigned the numbers 1 through 9. Position Ø is assigned 1Ø while position X is assigned 2Ø. By marking 2Ø, 1Ø, and 5 simultaneously, day 35 can be entered into the system. A blank column usually separates the day and point field.

Point Field:

Conferences with the Army have indicated that a point is usually labeled with a letter or letters followed by a number. These letters are G, I, P, R, S, GI, GP, GR, and GS where the G and I letters are simultaneously marked to get GI. A second column with same format as day field permits a number from 1 to 39 to be specified. A blank column separates the point field from the period field.

Period Field:

The period field can be coded in the same format as the day field

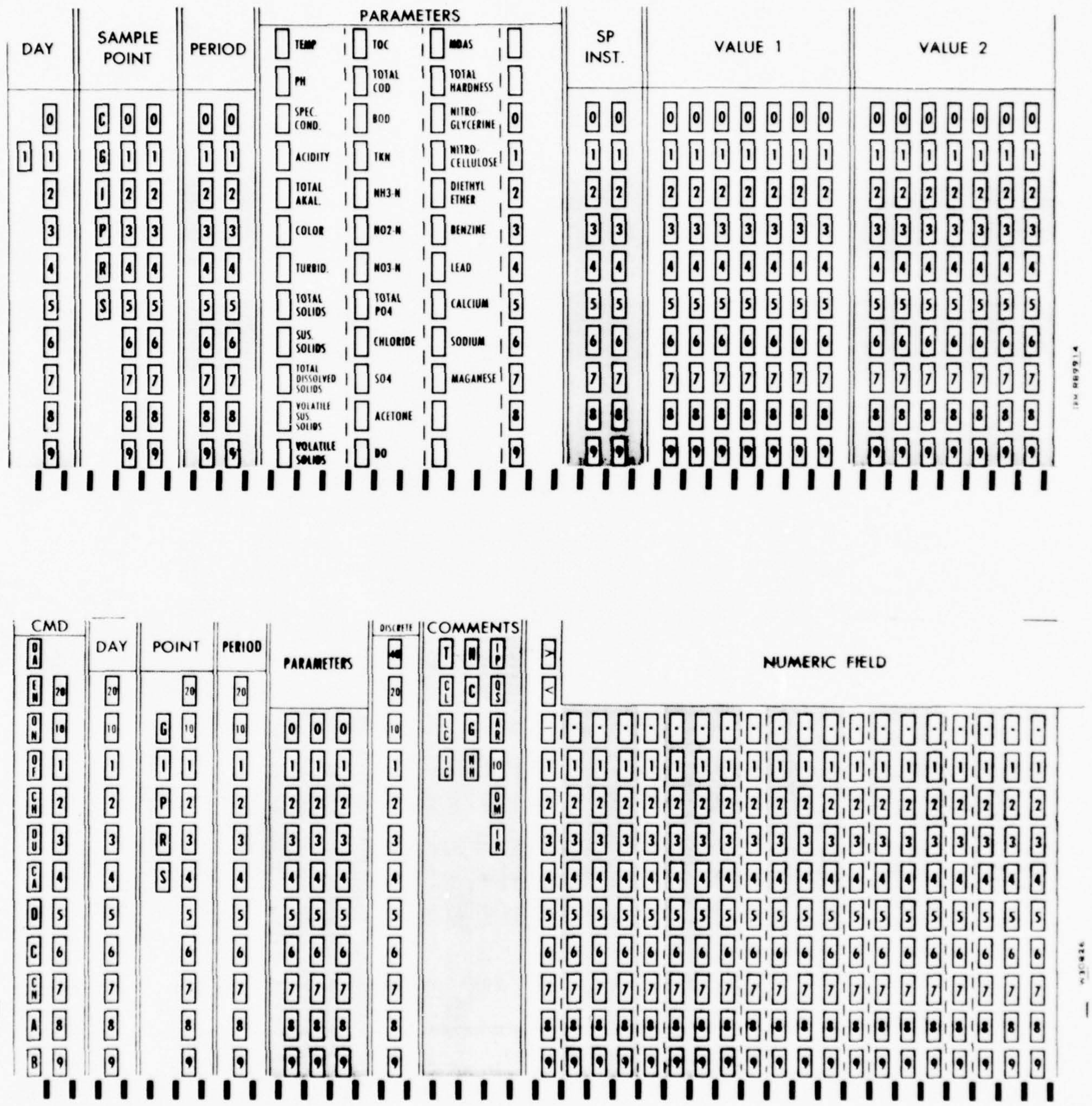


Figure 1 Examples of Mark Card Designs

to permit periods from 1 to 39.

Parameter Field:

The parameter is coded as an integer from Ø to 999. By assigning different numeric codes to the same parameter, the measurement method can be identified. Three columns are allocated to the parameter field for specifying the parameter code. An additional column separates the parameter field from the discrete field.

Discrete Field:

One column is provided for entry of the discrete ID. This format is the same as the day field except that the R punch position is used for marking 4Ø. This format handles up to 69 unique discrete values.

Quality Field:

The quality of a sample may be characterized by one or more of the following tags:

a) Turbid (T)

b) Clear (CL)

c) Light Color (LC)

d) Intense Color (IC)

e) Hot (H)

f) Cold (C)

g) Greasy (G)

h) Nonhomogeneous (NH)

i) Imperfect Preservation (IP)

j) Questional Stability (QS)

Note that three columns are required to code these tags. Since one or more tags can be marked simultaneously, three columns insure a resulting Hollerith character.

Accuracy Field:

The accuracy of the measurement may be characterized as follows:

a) Accurate Result (AR)

b) 10% (10%)

c) Order of Magnitude (OM)

d) Inaccurate Result (IR)

This information is combined with the quality ID information and placed within the same field.

Comment Field:

The comment field is combined with the numeric field. This first column in the numeric field permits a greater than or less than comment to be entered.

Numeric Field:

One or more measurements must be entered from the numeric field. One or more successive blank columns segment the information within the numeric field into one or more measurement values. The Ø punch should be coded as Ø while the X punch should be coded as the decimal point. In cases where the size of this field is inadequate, the program INPUT permits the numeric field to be continued onto additional cards.

Color of Card:

The color of the card selected to be compatible with the optical mark reader. The presence or absence of a mark is detected by the presence or absence of a reflection of light from the card surface. Usually, any black printing on the card is read as a mark.

Usually, every other field on the card should be shaded to assist the user in locating the desired field of interest. Adjacent subfields in the numeric field would be shaded differently.

Translation Tables:

The program INPUT which operates the card reader contains two levels of translation. During input, the following steps are followed by this program:

1. The card reader transmitts one character to the computer for each column on the card. The card reads each card using standard 029 character set.
2. The computer translates, with a translate table, each character received from the card reader into a character string. A unique translation table is present for each column.
3. This resulting character string is segmented into fields. Each field is translated with a unique translate table to yield a translated card image.
4. The input program always prints on the Teletype printer the translated card image. The input program also processes the translated card image.

Order of Fields:

The order in which the fields on the card appear may be changed. If this order is changed, the order of entries in the tables at PUTBH1, PUTBH2, and DEFI00 in the program CRH.PA must be altered. The numeric field should always be last.

OS/8 INDUSTRY COMPATIBLE MAGNETIC TAPE FORMATS

The subroutine NTRAN provides FORTRAN V programs written for the Army UNIVAC 11Ø8 with capability for reading and writing magnetic tape compatible with the OS/8 system. The subroutine NTRAN is well documented in the UNIVAC FORTRAN V Library Programmers Reference Manual (UP-7876). In addition to reading and/or writing binary information of tape, this subroutine performs input/output operations such as tape positioning. To use this subroutine, it is necessary to know OS/8 magnetic tape formats.

The OS/8 software normally performs 7-track magnetic tape read/write operations with an 800 BPI density and odd parity. A record consists of 256 6-bit tape characters with an even number of records per file. The OS/8 software packs three 8-bit ASCII characters within 4 consecutive 6-bit tape characters.

The PDP-8 is a 12-bit word oriented machine. The OS/8 software packs three 8-bit ASCII characters into two words as follows:

Word 1	Chracter 3 Bit 0-3	Character 1
Word 2	Character 3 Bits 4-7	Character 2

The parity bit associated with each character is ignored by the system and is usually "1". Transfers between core and mass storage are performed with 256 word blocks with each block being two 128-word records.

Let 4 consecutive tape characters be represented as follows:

$$\begin{array}{cccccc} A_0 & A_1 & A_2 & A_3 & A_4 & A_5 \\ B_0 & B_1 & B_2 & B_3 & B_4 & B_5 \\ C_0 & C_1 & C_2 & C_3 & C_4 & C_5 \\ D_0 & D_1 & D_2 & D_3 & D_4 & D_5 \end{array}$$

The three consecutive ASCII characters are as follows:

$$\begin{array}{cccccccc} A_4 & A_5 & B_0 & B_1 & B_2 & B_3 & B_4 & B_5 \\ C_4 & C_5 & D_0 & D_1 & D_2 & D_3 & D_4 & D_5 \\ A_0 & A_1 & A_2 & A_3 & C_0 & C_1 & C_2 & C_3 \end{array}$$

The parity bits A_4, C_4, and A_0 have no meaning but are usually "1."

The following ASCII control characters are commonly used within a file and are shown in octal.

CTRL/Z	(ASCII 232)	Logical EOF
Form Feed	(ASCII 214)	Skip to next page
Horizontal Tab	(ASCII 211)	Tab stop every 8 col.
Line Feed	(ASCII 212)	Normally ignored
Carriage Return	(ASCII 215)	Terminates each character string
Null	(ASCII Ø)	Ignored

All OS/8 software accepts variable length character strings as input and generate variable length character strings on output. Each character string is terminated with a carriage return. A character string frequently crosses record boundaries with numerous character strings per record. A carriage return on output is often treated by devices such as Teletypes as a carriage return/line feed combination.

The ASCII character sets are usually limited to those associated with a standard ASR-33 or ASR-35 Teletype.

The OS/8 program CAMP is useful for positioning the industry compatible tape prior to use. In addition, several CCL commands are provided for tape positioning.

The OS/8 Software Support Manual (DEC-S8-OSSMB-A-D) provides in Chapter 4 complete details on the operation of the TM8-E magnetic tape handler. Appendix E of this manual gives a complete list of character codes and conventions.

SORT

Introduction:

The program SORT retrieves selected data from a specified data file, sorts this data and then places the sorted data in an output file. Instructions for this sorting operation are received from an information file. Each record of this information file contains:

1. Sorting order
2. ID lists
3. Segment size
4. Sorting options

Each record of the information file defines a new sorting function which results in a new output file.

The SORT program may be loaded by

```
.R SORT
*(Output File) < (Info File), (Data File 1), (Data File 2)
```

When loaded, the operator must supply to the Command Interpreter

1. Initial Output file name
2. Information file name
3. First data file name
4. Second data file name (optional)

The extension of the initial output file is incremented to determine the name of successive output files. The last two characters of the output file name should not be "D" as the system uses this to denote a duplicate data file. Usually, the SORT option is one step in a BATCH command input file.

An END message will be printed at the completion of each sorting operation. When all the records of the information file are processed, control is returned to the OS/8 Monitor.

Normally, the software names the information file INFO:INFO.IN while the output files are named DATA:DATA.AA, DATA:DATA.AB, etc. The extension of the first output file is .AA but is incremented with each addition output file.

The SORT program checks to see if there is 2 or more points with identical Day, Point, Period, Parameter, and Discrete IDs present in the data file. The first measurement encountered with this ID will be used in the sorting operation. All remaining measurements with this ID are placed in a duplicate data file with a DUP.DATA message generated in place of the END message. If duplicate data were encountered while generating DATA:DATA.AF, duplicata data would be placed within the file DATA:DATADD.AF.

When the report sorting option is selected, the Day, Point, Period, Parameter, and Discrete (Discrete data only) IDs of this measurement be present in the ID lists provided by the information file to be selected for this sorting step. The plot sorting option additionally requires that either the Point and Parameter ID for continuous data or the Day, Point, Period, and Parameter ID for discrete data be found in the same relative location within the respective lists.

Theory of Operation:

The program SORT is a PAL8 assembly language program for which a save file can be generated with the following steps:

1. Assembly SORT.PA with PAL8 to obtain SORT.BN
2. Load SORT.BN with ABSLDR
3. Save location 0-3377 with a starting address of 200.

Both the parameter INFBSZ and the table TBUF may be modified with the text editor EDIT.

The SORT program consists of a main program which starts at location 200 and resides on this page. The remaining coding are subroutines and constants. The source of SORT is well commented to provide detailed documentation of its operation.

Each sorting function is accomplished with two steps. The first sort step retrieves from the input files each measurement which satisfies the ID lists provided by the information file and places these selected measurements in a core buffer. The second step sorts this buffer according the information file sort specification and then transfers this sorted core buffer to the output file. The size of the output file must be smaller than the core buffer. Duplicate data when found is placed in a duplicate data file.

During the first step, SORT requires all selected measurements be either discrete or continuous depend on the discrete ID sorting order. A zero discrete sorting order specifies continuous data while a non-zero value specifies discrete measurements. For all sort options, the Day, Point, Period,Parameter, and Discrete ID of the selected measurement must be present in the appropriate information file ID list. In addition, the plot option requires either the Point and Parameter or the Day, Point, Period, and Parameter ID to be present in these ID lists at the same relative location.

Field 0 of memory is dedicated to the program (0-3377), the various I/O buffers, and I/O device handlers. The remainder of memory may be allocated to the core buffer which holds the selected data from the input files. Memory (3400-7577) in field 0 is dynamically allocated by SORT for I/O buffers and device handlers. Dynamic allocation attempts to minimize the number of mass storage accesses.

Space is first allocated whenever required to the I/O device handlers used by the output file, information file, and two input files. The information file buffer size in blocks is determined by the parameter INFBSZ. The parameter INFBSZ has been arbitrarily set to 3. The remaining space is allocated for use by the input files. The output files use the same buffer space as the input files.

A table which starts at location TBUF specifies the various blocks of memory which may be used for the core buffer. Each block in this core as specified by constants which are:

1. CDF to field containing block
2. Starting address of block
3. Negative of number of measurement in block

The following example shows two blocks in this table.

```
TBUF,   CDF 20          / Field 2
        0000            /Starting Address
        -300            / Holds 192 Measurements
        CDF 10          / Field 1
        2000            / Starting Address
        -100            / 64 Measurements
        0               / end of list
```

Note that a zero terminates this list and each measurement requires 9 consecutive memory locations.

The User Service Routine (USR) is core resident and hence the locations (10000-11777) should not be used for the core buffer. SORT automatically allocates this space whenever the core buffer, defined by TBUF, overflows. If the buffer space provided by the USR overflows, then SORT treats this as a fatal error.

Overview of Subroutines

TBOFL

Allocate USR space to TBUF on overflow.

ININFO

Load handler for INFO file and initialize software to read this information file.

RINFO

Places the next record of information file in the I/O buffer starting at INFBUF. This may or may not require a physical read. Both the sort and search options for this record are also initialized.

CHK

This subroutine determines whether a given ID is in an ID table or not. If present, the subroutine determines which entry in the table. The starting entry within this table is specified by the calling program.

SETTR

The subroutine TRANSI is initialized with the first entry in TBUF table.

TRAN5

Update TRANSI subroutine pointers to the next entry in TBUF. If a TBUF table becomes entry, the subroutine TBOFL is called to allocate USR space.

TRANSI

Transfer one point from the input buffer to the output core buffer.

FIX

Fix a floating point number. The sorting order, segment length, and sort option are in a floating point format.

SEARCH

This subroutine is initialized by SEARIT to either the report or plot option and to whether the data is either discrete or continuous. Each measurement in the input data file is compared to the ID lists in the information file to determine whether this measurement has been selected or not.

SEARIT

This subroutine initializes SEARCH to the options specified with the current information file record.

NXID

A subroutine used by SEARCH to unpack ID or current measurement.

CNTRC

This subroutine checks to see if a control C has been typed. When received, SORT program is aborted and control returned to the monitor.

SETTRI

This subroutine initializes the subroutine TRANS to the first block specified by the table TBUF.

TRANS

This subroutine transfers a measurement from the core buffer to the output file.

GETBF

Allocate one block for use by I/O handler.

CHKBF

This subroutine checks to see if the block allocated was used by the I/O handler. If not used, return this block to the buffer pool.

OUTFL

This subroutine allocates one block for output handler, loads handler, and then enters this output file.

PUT

One block in output buffer has been filled. If output buffer is full, then write entire buffer out by calling PUTOUT.

PUTOUT

The entire output buffer is written to output device.

CLOSFL

Empty output buffer to output file and then close this output file. If this output file corresponds to last record in the information file, then the last unused point in this output data file is set to 1.

TOUTP

This subroutine checks the number of points on current block on the output file. If the block is full, the subroutine PUT is called.

BUFINT

A block in the output file I/O buffer is initialized by setting all entries to zero and setting up the block pointers.

INITIN

The I/O handlers for both input data files are loaded. The first record of the data file is read in input buffer.

REINP

Initialize for data file input.

IRINP

This subroutine is called by GTMORE and REINP. The input pointers to input file are initialized.

GTMORE

Set the input pointers to the second file.

GETNRI

Read the next input buffer. As several blocks are read with each physical read, this subroutine on many calls will change pointers to the next block in input buffer instead of performing a physical read.

SRCH

The core buffer is search for a measurement having a specified ID.

GDAY

Get the next valid day for the core buffer search.

GPAR

Get the next valid parameter for the core buffer search.

GPNT

Set the next valid point for the core buffer search.

GPER

Get the next valid period for the core buffer search.

SETSOR

Initialize the subroutine SETMSK to specify IDs consistent with sorting order and sort options.

SETMSK

This subroutine generates all ID tags in an order consistent with the search function for use with the subroutine SRCH.

GEXT

Get the next valid discrete ID for the core buffer search.

MSG

This subroutine prints messages.

FDOPT

This subroutine finds the options to be used during the sort and converts these sorting options from floating point to fix point notation.

CHGNM

When duplicate data is present, this subroutine converts the last two characters in the current output file name to DD.

DUPTR

If duplicate data is present, output these measurements to a duplicate data file.

FDINF

When SORT is chained to from another program, this subroutine is called to load the handler INFO and open the file INFO.IN for input.

FDOUTF

The output file handler is loaded. When chained, the output file is given the name DATA:DATA.AA.

GETHDL

A handler is loaded.

CHAIN

When SORT is chained to from another program, the SORT program chains to the program REPORT or PLOT. This option is valid for older versions of the data handling software.

FNCTN.AR

Each data reduction algorithm is implemented using PAL8 and is a subroutine within the OS/8 ASCII file FNCTN.AR. The present file contains 14 subroutines with the names FXXØØ, FXXØ1, ..., FXX13. Each algorithm is a function of one or more measurement values, user supplied constants, and one or more piecewise linear functions. This algorithm definition must agree with the file METHOD.AR description in terms of a 2 digit identification code, number of measurements, number of user supplied constants, and number of coordinates for any piecewise linear function.

There are several rules which must be followed when adding or modifying algorithms. These rules are as follows:

1. The subroutine resides in field 1 of the PDP-8/E memory.
2. The subroutine name must be 5 characters with the first 3 characters being FXX and the last 2 characters being the algorithm identification code. This identification code is a positive integer.
3. The subroutine is called from field Ø with the return being back to field Ø.
4. The AC when called is an address in field 1 of a buffer containing user supplied constants and (X,Y) coordinates for piecewise linear function.
5. When called, the calling program has already determined that all data required by this algorithm is valid.

6. The measurement value computed by this subroutine is placed in the floating point AC in field Ø prior to return.
7. The measurement values are in a buffer starting at TMPBFR in field 3. Each measurement is in floating-point notation with three words per variable.
8. The floating-point package is available for use in field zero.
9. The user supplied constants are in floating-point notation when present and are placed in the buffer prior to (X,Y) coordinates.
10. The subroutine FNCTX in field Ø can be used to evaluate any piece-wise linear functions. The calling sequence is

```
CIF Ø
JMS I (FNCTX)
Table Address in Field 1
Address of X in Field 1
Return
```

with the value of this function in the floating point AC.

11. If an error is detected during the computation, the subroutine should increment the return address to announce this error.

BASUF.PA

The OS/8 ASCII file BASUF.PA contains six BASIC user functions. This file is written using PAL8 assembly language and is consistent with the OS/8 BASIC Run-time System (BRTS). When assembled, loaded, and saved, the resulting file can serve as BASIC.UF. The file BASUF.PA is well documented through the use of numerous comments. However, this documentation requires a knowledge of the material in Chapter 6 of the OS/8 Handbook relating to assembly language functions for BASIC.

These user functions provide BASIC with the following capabilities:

1. Unpack ID tags (UFNC)
2. Pack ID tags (UFNC2)
3. Accept PDM-70 input (TTYCHG)
4. Formatted output (OUTPUT)
5. Interize (INTEGE)
6. Chain (CHAIN)

The last user function is no longer used. The BRTS has been patched to provide the entry points 34ØØ, 3423, 36ØØ, 42ØØ, 3511, and 4ØØØ for these user functions.

The formatted output and the modified Teletype handler user functions make special use of the BRTS routines which are not documented in the OS/8 Handbook. The modified Teletype handler user function inserts a new copy of Teletype handler at location 6412 when receiving input from the PDM-7Ø. This handler at 6412 is converted by this user function to its original state when receiving input from the Teletype.

The output function uses the BRTS code at 2ØØØ - 2Ø3Ø to specify a unit number for I/O. The function has the subroutine SETUN which performs the following operations:

1. Change location 2Ø3Ø from a 5513 to 56ØØ.
2. Calls this code at 2ØØØ with JMS I (2ØØØ) and file number in AC.
3. Restores location 2000 to 4514 and location 2Ø3Ø to 5517.
4. Sets up pointer at XPUT output routine.

For output on file numbers 1 through 4, the subroutine PUTCH in BRTS is called to place the 8-bit character in AC in the devices output buffer. For file Ø, the subroutine at 1ØØØ is called to place a character in the printer buffer.

DAHDLR.BA

The OS/8 BASIC program DAHDLR.BA is generally used for the report specification steps and provides the capability of creating information files, listing information fiels, listing data files, and creating a batch file to perform a sort and report operation. The program asks the user for a function to be performed and the branches to the appropriate section of the program to do this function. Each function may request and accept additional information from the user.

Each record of an information file defines either a report or a plot. Each record contains the sorting order, option, segment size, and ID lists. When listing a record, this information is converted to an external format and printed. When creating, this program contains an editor which can add, change, list, or delete entries within the various lists within each record of the information file. When these lists for a given record are valid, they can be output.

The following identifies the operations performed by the various statements within the program.

20-95	Initialize program constants and input the system definition file.
100-155	Input function from keyboard. If valid, then branch to the appropriate function.

LIST

704-850	List the measurements within a specified data file. The variable P9 determines whether the secondary ID tags are listed or not.
705-715	Get name of the input file and open the input file.
720-765	Get the next measurement and the primary ID tags and measurement value.

766-850	If P9=1, then print secondary ID tags for this measurement.
<u>LISTPRI</u>	
680	Set P9=1 and do a list function
<u>LISTDS</u>	
600-660	Get name of ID list from user and then list all entries in the list. The number is the internal ID while the character string is the external ID tag.
600-620	Did we get a valid input?
625-640	Print chemist list.
650-660	Print other lists.
<u>LINFO</u>	
3400-3403	If a file is open on #2 then close it. Open the file INFO:INFO.IN. Set to record Ø.
3404-3408	Read a record and print. If more records, repeat until all records are printed.
3410-3520	Subroutine to read the next information file record. If present, then print this information on the printer.
<u>LISTINF</u>	
575-585	If the file on #2 is open, then close it. Get the name of this information file and branch to line 3403 in the LINFO coding.
<u>STR</u>	
500	Set Y8 = -1
<u>NSTR</u>	
510	Set Y8 = 1

REPORT

250-465	Ask the user whether a report or plot is desired. Ask user whether continuous or discrete data. If a report, ask user for sorting order.
200-210	Prompt user with (#) to receive a command. If non-string option, go to line 3600 for input. If string option, receive input command string and go to line 400 to decode this string.

REPORT (NSTR Option)

3600-3610	Set all fields K1$(I) to null strings before input.
3615-3650	Input type of command K1$(Ø). If this completely specifies command, go to 3645 and execute. Otherwise, get remaining fields of this command string.
4130-4175	Is K1$(1) and ID code is DAY, PNT, PER, PAR, or DIS. If yes and this is a new ID, then save last ID list and restore list for this new ID. K2$ is name of last ID.
4185-4195	Remove K1$(1) if ID name from command string.
4200	If a plot (Z8 = 1), go to line 6400 to process plot command.
4201-4225	Which ID name is being processed. If there are no ID tags given in the input, then go to line 4900 and decode the command.
4226-4260	Input command gives a list of ID tags. Convert these tags to internal format. If ID error, notify user and get new command.
4900-5090	Decode the command which is given in K1$(Ø) and branch to the appropriate program to perform the indicated operation.

4300-4330	Subroutine to unpack the ID list specified by J and place in array C.
4350-4375	Subroutine to pack the array C into the ID list specified by J.
4400-4435	Subroutine to delete the ID with code L3 from the list C.
4450-4470	Subroutine change the first occurrence of the ID L3 in the list C to L4.
4500-4560	Subroutine to insert the ID L4 in the list C before L3.
4600-4710	A standard report (STD) has been requested. Ask user for record number. If legal record, read this record and make all lists agree with this record.
5100-5140	List (L) an ID list.
5200-5250	Add (A) one or more entries to an ID list.
5260-5295	Delete (D) one or more entries from an ID list.
5300-5495	Change (C) an entry in a list.
5500-5525	Delete (NEW) all entries in the ID list.
5600-5605	Print "Command Error".
5700-5720	Set input option to STR or NSTR.
6000-6005	If new data type, go to line 6100.
6100-6180	Get order of sort from user (LORD).
6025-6045	Get segment (SEG) size from user or list segment size (LSEG).
6060-6066	Print sorting order (LORD).
6075-6080	Close information file (END).
6200-6395	Output (DO) current lists if correct to the open information file.

6400-6940	The plot commands (AP) and (LP) are implemented here. For continuous data, PNT and PAR must be added to list. For discrete data, the DAY, PNT, PER, and PAR must be added to the list.
7000-7025	Clear (NIDS) the current ID list.
7050-7060	List (LTAB) the status of current tables.

REPORT (STR Option)

400-4120	Segment the command string B2$ into fields. Each field is separated by a space. Once the fields K1$(I) are known, the coding is the same as for NSTR option.

PLOT

400-465	The plot command does the same as report. The tables are set for plot option and plot sorting order.

PRESTD

350-355	The operation is same as REPORT except that the name of information file is INFO:STD.IN.

PREINF

550-565	The operation is same as REPORT except that the name of the information file is specified by user via the keyboard.

DOREPORT

1300-1385	List the names of the files given in DSKDIR.AR and SYSDIR.AR. Receive the name of the master file from the user. Create a Batch job to do a report with the data coming from this master file and the information file created by REPORT.

GENERAL SUBROUTINE

1200-1235	This subroutine gets the name of an output file and opens this file on #1.

1250-1290	This subroutine gets the name of an input file and opens this file on #2.
1400-1485	This subroutine inputs the short ID tags from the SYSDEF.AR file.
1500-1535	This subroutine converts the external ID to an internal ID.
1600-1660	This subroutine inputs a measurement from the data file on #2. If the end-of-file is reached, close this file and print the number of points found.
2400-2465	This subroutine outputs the sort order, ID lists, and options to create one record in the information file.
2500-2540	This subroutine skips one record of an information file.
2600-2915	This subroutine reads one record of the information file.

DAHDLR Variables

Z8 = 0	Report Option
Z8 = 1	Print Option
Z9 =	Segment Size
L6 = -1	Info file not open for input
$\geq$ 0	The next record of information file to be read on input.
Y8 = 1	NSTR input option
= -1	STR input option
K2$ =	Current ID list (DAY, PNT, PER, PAR, DIS).
K3$ =	Current data type CONT, DISC)
J =	Current ID list

0 = Day
1 = Point
2 = Period
3 = Parameter
4 = Discrete
5 = Comment
6 = Quality

	7 = Accuracy
D(I) =	Sorting Order
CI(I) =	Packed ID lists
C(I) =	A single unpacked ID list
A$(I) =	Array contains short external ID tags
A4$(I) =	Array contains Chemist external ID tags
Y9 =	No information file open for output
≥ 0	Which record in open information file on output
KI$(I) =	Field I in command string
AI$ =	Name of output file
AZ$ =	Name of input file

IDGEN.BA

The Basic program IDGEN.BA accepts as input the files

DSK:TRNTBL.AR

DSK:SYSDEF.AR

DSK:METHOD.AR

and produces the temporary OS/8 ASCII files

DSK:SYSDEF.PA

SYS:PARCRH.AR

SYS:CVRTBL.AR

The file SYSDEF.PA when assembled by PAL8 with the F option and loaded become the ID tables and translation tables for the program INPUT.SV. The file PARCRH.AR is a parameter definition file which must be used when the files CRH.PA, GETPUT.PA, FNCTN.AR, and CVRTBL.AR are assembled by PAL8 with the F option. The file CVRTBL.AR are the parameter conversion tables for the program INPUT.SV which specifies the conversion algorithm for each parameter on input.

The program INPUT.SV contains several tables which are application dependent. The length of each table is highly variable. The program IDGEN.BA generates many of these tables and dynamically determines the starting address of each table to effectively utilize memory. All of the files generated by IDGEN.BA are used as input to PAL8 with the binary output, when loaded with ABSLDR, become tables for INPUT.SV.

The statements within IDGEN.BA perform the following operations:

100-260	Define strings

300-405	Read from the file SYSDEF.AR the short ID tags for the Day, Point, Period, Parameter, Discrete, and Comment ID into S$.
410-532	Read from the file SYSDEF.AR the short ID tags for the Chemist ID into S1$ and compute the length.
535-575	A subroutine to convert 12-bit positive integers into octal.
579-592	Read from the file SYSDEF.AR the short ID tags for the Quality and Accuracy ID into S$.
600-900	Convert the short ID tags to tables and output to the file SYSDEF.PA.
910-927	Read in the OS/8 ASCII file METHOD.AR and generate the OS/8 ASCII file CVRTBL.AR.
930-945	Output to the file PARCRH.AR the parameter definition for the variables DAYADR, POINTA, PERADR, EXTRAD, COMADR, QUALID, ACCID, and CHEMAD.
947-960	Compute starting address (CVRTBL) for the file CVRTBL.AR and output to PARCRH.AR. The buffer constants for INPUT.SV are placed in the file PARCRH.AR by calling the subroutine at 3000.
1000-1195	Input the translation tables from the file TRNTBL.AR and compute length of tables for first level of translation.
1200-1231	Output to starting address (LTRAN) fo first level translation tables to PARCRH.AR.
1235-1275	Output first level translation tables to the file SYSDEF.PA with a format acceptable to INPUT.SV.
1300-1490	This is a subroutine called at 1267 to output first level translation tables.

1600-1730	Compute lengths of secondary translation tables.
1735-1845	Output input tables for the secondary translation tables to SYSDEF.PA.
1850-1931	Output output tables for the secondary translation tables to SYSDEF.PA.
1932-1960	Output to PARCHR.AR the starting address for the Format (FORMAT) and Command (CMDADR) tables used by INPUT.SV.
2000-2075	Subroutine called at 1930 to output to SYSDEF.PA the output tables for the second level of translation.
2100-2790	Subroutine, called at 920, to output the parameter conversion tables for which a method is specified in the METHOD.AR file.
2800-2855	Subroutine, called at 925, to output the parameter conversion tables for which a method is not specified in the METHOD.AR file.
3000-3075	A subroutine, called at 947, to output fixed buffer constants used by INPUT.SV to PARCRH.AR.

MERGE.BA

The OS/8 BASIC program MERGE.BA merges each data file created during data entry into a master data file. This program lists the name of each data file which has not been merged and receives a response from the supervisor on the particular action to be taken. The capability is also provided to either delete measurements or change measurements within the master file.

The OS/8 ACSII files DSK:DSKDIR.AR and SYS:SYSDIR.AR must be created by DIRECT with the E option prior to running this program. These files contain the names of the master files and data files. The MERGE program lists the names of each data file and requests from the supervisor the action to be taken for each file.

The operations performed by the statements within this program are listed as follows:

20-75	Initialize and print instructions to operator.
2000-2182	Open the files DSK:DSKDIR.AR and SYS:SYSDIR.AR for input. For each file, print the data of creation. Print the name of each file if not a master and ask the operator for an action. The action specified places the file name in the appropriate list. The extensions of the master files are compared to find the last master.
3300-3325	Get the names of other files and repeat many of the steps in lines 2000-2182.
2184-2190	Print the name of the last master file. Input the names of the next master output file and the next master input file.

150-250	A1$(2) is the name of the deletion file. If file is present, read the measurements within this file into the array A1.
300-350	A1$(3) is the name of the change file. If present, add these measurements to the array A1.
400-415	A1$(0) is the name of master output file. Open this file for output.
420-450	A1$(1) is the name of the master input file. Open this file for input.
450-605	Input a measurement from the input file. Compare this entry (Day,Point, Period, Parameter, and Discrete ID) to the entries within the deletion and change lists for a match. If not found, transfer this measurement to the output file. Otherwise, perform the indicated operation and update the array A1.
650-680	A1$(Q9) for Q9 = 4,5,... are names of data files. Transfer the measurements within these files to the output file.
700-725	Close the output file.
730-1200	If the deletion and change list now empty? If not, write these entries into a file DSK:UPDATE.ER and print error message to the operator.
1200-1210	Create a batch file to change the names of all files used as input by this program.
1400-1620	Subroutine to create a batch file to change the names of all input files. The first character of each file name is changed to the letter A.
800-840	Subroutine to output one measurement to open output file.

1000-1060	Subroutine to input one measurement from the open input file.
1100-1180	When a match is found between the input measurement and an entry within change and deletion list, this subroutine removes this measurement from the change and deletion list.
3200-3270	Subroutine is used to remove spaces from file names.
2400-2640	This subroutine receives from the operator an action command and places the name of this file in either an add list, deletion list, change list, or no action list.
2300-2350	This subroutine checks the name of each file to see if it is a master file. If yes, the extension is checked to see if it is the latest master. Otherwise, the subroutine at 2400 is called to receive operator action.

REPORT.BA

The OS/8 BASIC program REPORT.BA generates both reports and print-plots of selected data using a wide variety of report and print-plot formats. The program REPORT.BA receives as input the information file INFO:INFO.IN, the system definition file DSK:SYSDEF.AR, and the sorted data files DATA:DATA.AA, DATA:DATA.AB, etc., and places all output within the OS/8 ASCII file OUT:OUTPUT.RP Each record of the INFO:INFO.IN file specifies either a report or a plot with the data from the corresponding data file. The file SYSDEF.AR provides to the program the internal to external ID conversion, header information, parameter print formats, and maximum-minimum parameter values.

The secondary ID tag information is ignored by REPORT.BA. The report option and plot option use separate coding. The following material shows the purpose of each line or group of lines in the program.

190-220	K8 = Information file record number. Open files INFO.IN, SYSDEF.AR, AND OUTPUT.RP
230-330	Define string constants. C9$(I) = Tag for Ith Comment ID P$(I) = Plot I
340-400	Read next record of INFO.IN file. If EOF is reached, then end of report. Otherwise, this record defines B(I) = Sort order C(I) = Packed ID lists Z9 = Segment size Z8 = Mode (0 = report, 1 = plot) Open the next data file for input.

410-460	If a report (Z8 = 0) and continuous data (B(4) = 0), then make it appear to be discrete data with the Discrete ID being first order.
470	Increment INFO.IN file record number.
470-476	Compute the position of header when more than 5 columns are present.

Plots

640-690	Set M6(I) = 0
700-730	Set V(I) = Parameter ID of plot I Set VI(I) = Point ID of plot I
740-890	Input a measurement from the data file and find which plot it belongs. Then update M1(I) = Maximum of plot I M2(I) = Minimum of plot I M5(I) = Average of plot I M6(I) = Number of points in plot I
900-990	Unpack from C(I), the Day, Point, Period, and Parameter ID and place in B3(J).
1000-1320	Read from the System Definition File for continuous data the column ID for day and period and the long ID for point and parameter. For discrete data, read the long ID for the Day, Point, Period, and Parameter ID and the column ID for the discrete ID.
5990	Branch to either the continuous or the discrete data plotting routine.
6000-6185	Continuous Data Plot

7000-7155	Discrete Data Plot.
3324-3340	Bottom of Page and Page Advance.

<u>Report</u>

1390-1720	Read in 24 character long ID from SYSDEF.AR for 1, 2, 3, and 4 order sort ID. For 5th order sort, read in column ID.
1440-1480	Long ID read and not parameter.
1500-1540	Column ID read and not parameter.
1560-1620	Column ID read and format if parameter.
1670-1710	Long ID read and format if parameter.
1730-1783	If first record of INFO file, then read header into H$ from SYSDEF.AR.
2040-3240	Five nested FOR loops are present.
2090-2150	For YØ = Ø to 65 Unpack YØ ID from first order list. This ID is IØ. If IØ = Ø, then end of list is reached provide YØ is not zero and B(4) is not one.
2190-2320	For Y1 = Ø to 65 Unpack Y1 ID from second order list. This ID is I1. If I1 = Ø, then the end of list is reached.
2360-2450	For Y1 = Ø to 65 Unpack Y2 ID from third order list. This ID is I2. If Y2 = Ø, then the end of list is reached.
2470	Intialize column and row unpacking pointers.
2480-2520	Are there any measurements for this page which have ID tags IØ, I1, and I2. If not, skip this page.

2530-2720	Print header for this report.L9 = line counter for page. B1$(I¢) = long ID for IØ. B1$(I1 + 64) = long ID for I1 B1$(I2 + 129) = long ID for I2
2730-2740	Unpack the next column ID tags for this page of report. B5(I) is ID for column I. If list is empty, then end of page.
2760-2950	For Y3 = 0 to 62 Unpack Y3 ID from fourth order list. This ID is I3. If I3 = Ø, then end of list is reached. This ID identifies a row in the report. If no measurements for this row, then skip this row. B1$(I3 + 194) is long ID.
2960-3150	For Y4 = 0 to 28. Print the Y4 column in this row of the report. If measurement is present, it has an ID of I4.
3160	Are there more columns to be printed for this page?
3170-3240	End of FOR loops.
3250-3340	Advance page and prepare for next report.

Subroutines

3360-3430	Unpack next ID for report line.
3440-3560	Unpack next set of ID tags for report columns. The number of ID tags unpacked is determined by segment size.
3570-3610	Skip record in SYSDEF.AR file.
3620-3840	Read next measurement from data file. B = measurement value A = 0 if end-of-file

	TØ = First order ID in measurement
	T1 = Secondary order ID in measurement
	T2 = Third order ID in measurement
	T3 = Fourth order ID in measurement
	T4 = Fifth order ID in measurement
	T5 = Comment ID
	T9 = Format of measurement
	T6 = 1 Point if read
3850-3930	Label columns in report.
3940-3980	Print (-----) to separate pages in report.
3990-4030	Page advance.
4040-4100	Compare measurement ID tags (TØ, T1, T2, T3) with expected ID tags (IØ, I1, I2, I3). V1 = 1 if disagree and Ø if match.
4110-4150	Print report header.
4160- 4280	Compute print position for plot Y2.
4300-4420	Unpack an ID character string from B1$ and place in B2$. Ø4 gives number of elements in each entry while Ø2 gives entry. If parameter ID, get plot maximum and minimum.
4430-4480	Get next ID list from SYSDEF.AR file.
4500-4530	Print series of (-) to identify the start and end of plot.
4540-4760	Print one line of a plot.

TDSINF.BA

The OS/8 BASIC program TDSINF.BA creates the information file INFO: TDSINF.IN. This information file when used with SORT and a master data file creates two sorted data files which serve as input to the program TDS.BA. The program TDS.BA computes TDS at a given sample point from the TS and SS measurement at this sample point. This information file has two records with the first record being for continuous data while the second record is for discrete data.

The operations performed by the statements within this program are as follows.

100-250	Open the file SYSDEF.AR and find the internal ID tags for the TS, SS, and TDS parameters. Count the number of ID tags in each list.
300-320	Generate the first record of the file INFO:TDSINF.IN for continuous data with parameter as the fourth order sort.
330-360	The second record of TDSINF.IN for discrete data is generated with parameter as the fifth order sort.
400-430	Subroutine to create list of ID tags for Day, Point, and Period ID tags. Each list includes all the tags in the system definition file. The subroutine at line 500 is called.
475-495	Subroutine to place TS, SS, and TDS ID in the parameter list.
500-590	Subroutine to create an ID list which contains all ID tags.

600-650	Subroutine to convert an external ID to an internal format.
700-797	Subroutine to create one record in the information file.

TDS.BA

The OS/8 BASIC program TDS.BA performs the total Dissolved Solids (TDS) computation. This program receives input from a user specified sorted data file and places its output in a user specified data file. For each Total Solids (TS) measurement, the next two measurements, when present, are the Suspended Solids (SS) and the Total Dissolved Solids (TDS) measurement with the same Day, Point, Period, and Discrete ID tags. If both the TS and SS measurements are present and the TDS measurement is not present, then the program computes the TDS measurement and places it in the output file.

The continuous data within this file should have the parameter as the fourth order sort. The discrete data within this file should have the parameter as the fifth order sort.

The following classifies the BASIC program statements according to the operations they perform.

20-120	Open the file SYSDEF.AR and find the internal ID tags for the parameters TS, SS, and TDS.
122-155	Get the name of the sorted input file and the name of the output file via the keyboard from the user. Open these files.
200-235	Input a measurement. If TS, go to statement 300. Otherwise, go to statement 200.
300-365	Input a measurement. If SS and if its ID agrees with last TS measurement, then go to 400. Otherwise, go to 210 and see if it is a TS measurement.

400-535	Input a measurement. If it is TDS and its ID agrees with the last TS and SS measurement, then do not compute TDS and go to line 200. Otherwise, compute TDS and return to 210.
600-630	Finished. Close the output file.
1000-1055	Subroutine to input one measurement.
1100-1125	Subroutine to output one measurement.

CRH.PA

The program INPUT.SV is created by loading the following files with the absolute loader ABSLDR and saving the core image.

File	Function
NONEAE.BN	Floating Point Package
CRH.BN	Major part of INPUT.SV
SYSDEF.BN	ID and translate table
CVRTBL.BN	Function subroutines and conversion tables
GETPUT.BN	Subroutine to pack and unpack OS/8 character buffers

The core maps for each 4K field of memory are shown in Figures 1, 2, 3, and 4.

The major portion of the program CRH.PA is that portion of INPUT.SV which remains constant while that portion of INPUT.SV which implemented by SYSDEF.BN and CVRTBL.BN is highly dependent on the application. The files NONEAE.BN and GETPUT.BN are subroutines called by CRH.BN.

The program IDGEN.BA receives as input the files TRNTBL.AR and SYSDEF.AR and created the file SYSDEF.PA. This file when assembled by PAL8 with F option is SYSDEF.BN and contains the valid ID tag lists and the translation tables.

The program IDGEN.BA also receives as input the files METHOD.AR and SYSDEF.AR and creates the files PARCRH.AR and CVRTBL.AR. The file PARCRH.AR is a parameter definition file which defines the starting address of several tables. The file CVRTBL.AR contains tables which define the data reduction algorithm for each parameter. The binary file CVRTBL.BN is created with

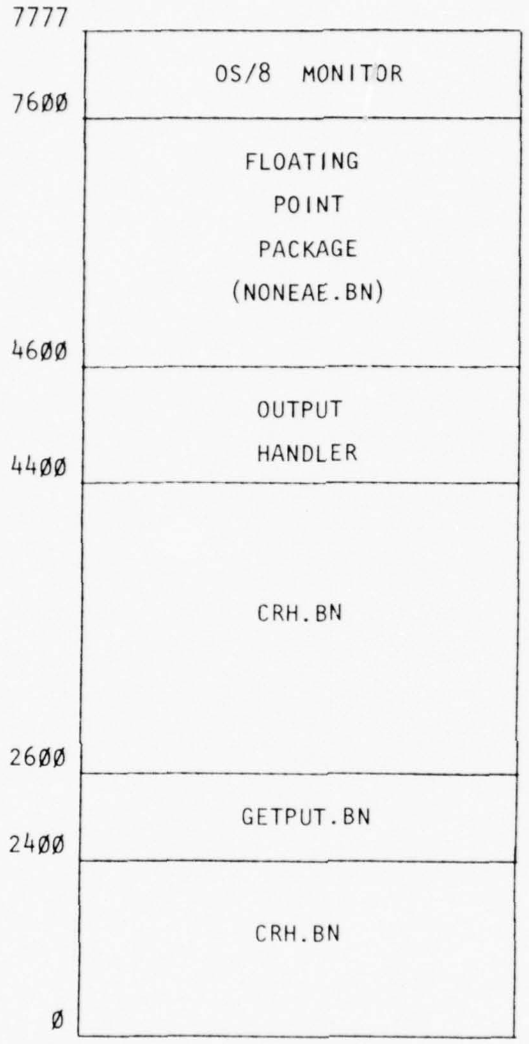


FIGURE 1 FIELD Ø CORE MAP OF INPUT.SV

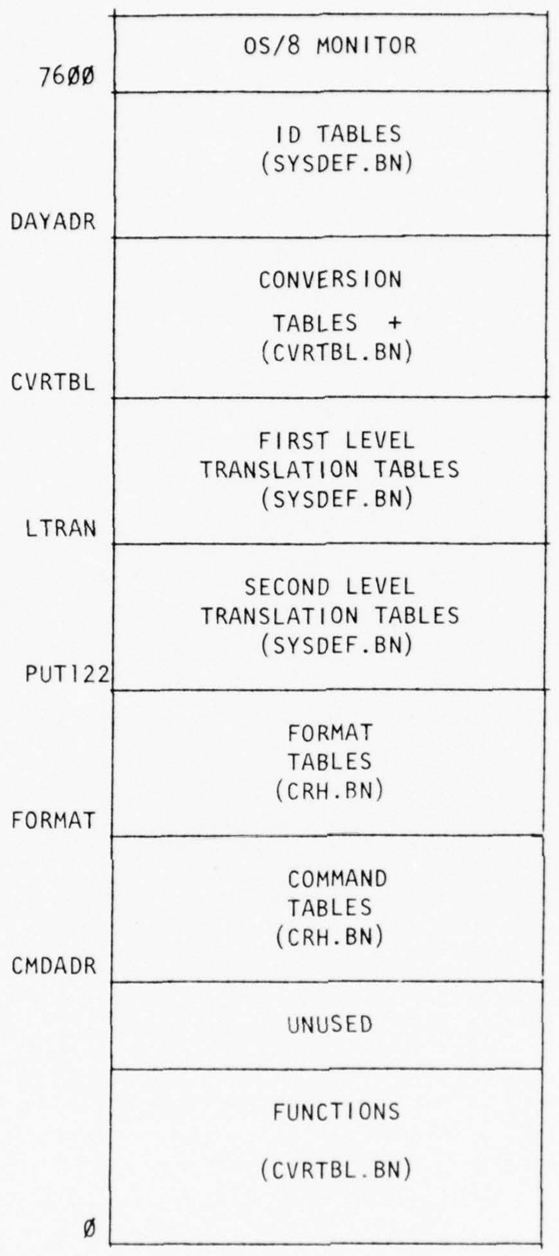


FIGURE 2 FIELD 1 CORE MAP OF INPUT.SV

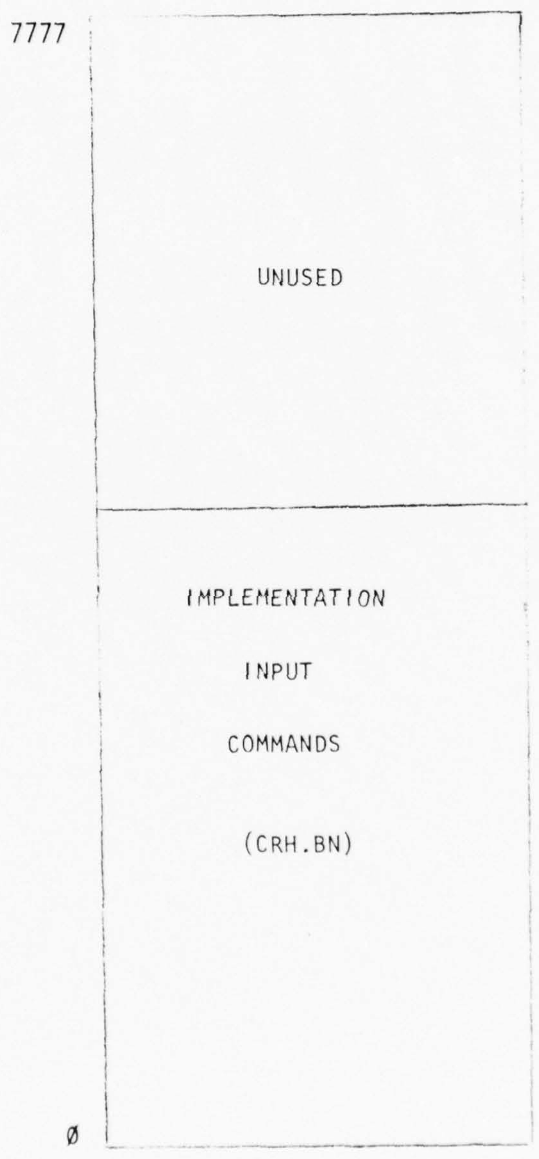


FIGURE 3 FIELD 2 CORE MAP OF INPUT.SV

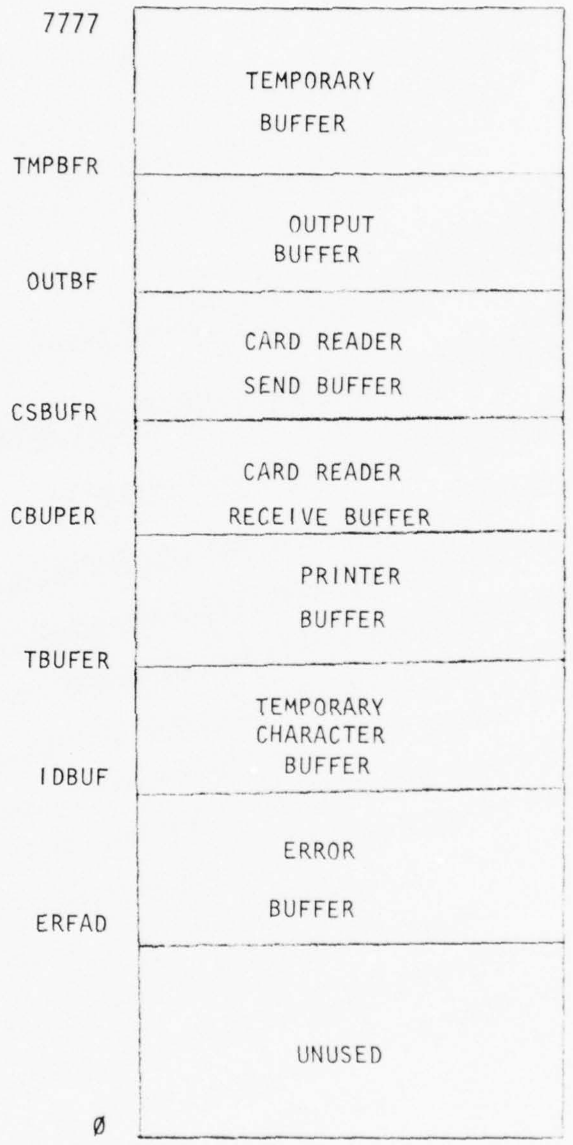


FIGURE 4 FIELD 3 CORE MAP OF INPUT.SV

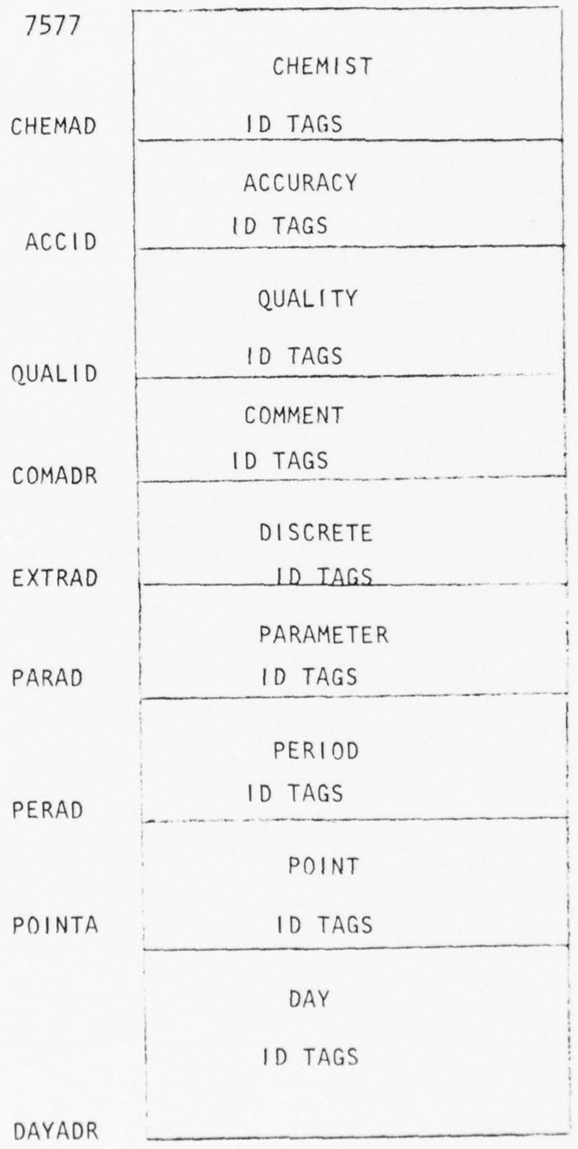


FIGURE 5 FIELD 1 ID TABLES

PAL8 using F option from the input files PARCRH.AR, FNCTN.AR, and CVRTBL.AR. The file FNCTN.AR contains several data reduction subroutines.

The binary files CRH.BN and GETPUT.BN are created with PAL8 by assembling CRH.PA and GETPUT with the parameter definition file PARCRH.AR. The file PARCRH.AS is required to specify the starting addresses of several lists to these programs which have been dynamically assigned by IDGEN.BA.

<u>OVERVIEW OF CRH.PA OPERATION:</u>

Each command line received on input consists of a character string terminated by a carriage return. For card input, each character must be translated into variable length character string with a unique translation table for each column. The resulting character string which results after translation is segmented by the presence of a space character into the folowing fields:

Command

Day

Point

Period

Parameter

Method

Discrete

Quality

Accuracy

Comment

Numeric

The presence of spaces in the numeric field segment the numeric field into measurements. The character string which defines each field is translated with a second level of translation to define a new character string for this field. All translation errors are recorded in an error buffer.

Tables which contain all valid entries for the Command codes and the Day, Point, Period, Parameter, Method, Discrete, Quality, Accuracy, Comment, and Chemist ID tags are present in memory. The character string for each field is compared to the entries in the appropriate table to determine whether this is a valid Command and ID tag. Invalid Command and ID tag are recorded as an error in the error buffer.

Valid Command and ID tags are converted from the external format of a character string to an internal format. The internal format is a positive unsigned 6-bit integer. A zero internal code indicates the absence of a valid Command or ID tag.

The translated card image is printed on the printer as the translated card image is considered to be the actual input to the program. When input comes from the Teletype keyboard, no translation takes place. A question mark is printed whenever a character can not be translated.

The internal Command code for the Command string is placed in location CMDCOD. An invalid Command will result with an internal code of -1. The ID tag codes for each command string are packed into locations WD1, WD2, WD3, WD4, WD5, and WD6. The character string which defines the numeric field has been modified to eliminate leading spaces, multiple spaces, and trailing spaces with the resulting string stored in the buffer starting at IDBUF.

The program now examines CMDCOD to determine whether a command or measurement was specified. A measurement has a null Command code. The routine FDISP determines the starting address for a function program to handle the specified command and then branches to this function routine.

For command line corresponding to a measurement, the subroutine DOCONV is called. The character string which defines the numeric field is converted with the resulting measurement values placed in the buffer TMPBFR. The number of measurement values found must agree with the number of measurement values expected. Otherwise, an error is detected.

The subroutine DOCONV calls the data reduction routine corresponding to this Parameter and Method ID. The data reduction computation is performed with the result being placed in the floating point AC. The result of this computation is printed on the printer.

For measurement lines, the ID tags packed in locations WD1, WD2, WD3, WD4, WD5, and WD6 are checked to see if these tags define a valid set. The Day, Point, Period, Parameter, and Chemist ID tags must be nonzero. The Discrete ID tag must agree type of data being received. Any errors are recorded in the error buffer.

If no errors are encountered, a valid measurement is temporary saved. This measurement is transferred to output buffer whenever the next command string is a measurement command string. This measurement must be saved in this manner as the entry of duplicate data is permitted with the final measurement value being the average of these measurements.

After each command string has been processed, the contents of the error buffer are printed to list all errors found. Only the first 10 errors are listed.

This program generally runs with the interrupt system enabled to permit concurrent I/O and computation. Ring buffers are provided for input from either the card reader or teletype keyboard, output to teletype printer, and the output of control characters to the card reader. Subroutines are provided to place characters in these ring buffers and to remove characters from these ring buffers. Each ring buffer is defined by a starting address, ending address, number characters in the buffer, the address of next free location, and the address of next character to be removed.

The following commands have been defined for the field Ø routines to transfer characters to and from the ring buffers.

PUTTP	Place character in AC in printer ring buffer. The AC is unchanged.
PUTCR	Place control character in AC in card reader ring buffer. The AC is unchanged.
GETCR	Get character from input ring buffer and place in AC.

Routines in field CUR4 use the subroutine PUTP2 to place characters in the printer buffer.

All transfers between mass storage and the program are performed with standard OS/8 handlers. Prior to using any OS/8 handler, the interrupt is disabled whenever the ring buffers reach a certain state. The interrupt is enabled after the transfer has been completed.

ID Tag Format:

The Day, Point, Period, Parameter, Discrete, Accuracy, Quality, and Comment ID lists contain the short ID tags. Internally, each ID tag is represented by six 6-bit ASCII characters which are packed into three consecutive words. If a given tag is less than 6 characters, the tag is padded with the (@) character. An ID tag for which all 6 characters are the (@) character terminates this list.

The program IDGEN.BA creates the file SYSDEF.PA which contains these ID lists. Some typical lines in this file relating to these ID lists are as follows:

```
DAYADR, TEXT  ?1@@@@@?
        TEXT  ?2@@@@@?
          .

          .

        TEXT  ?14@@@@?
        TEXT  ?@@@@@@?
POINTA, TEXT  ?S1@@@@?
          .

          .
```

The Chemist ID list is coded slightly different. Each short Chemist ID is padded with either one or two (@) characters to yield a character string which is even in length. This list is terminated by two successive words containing (@@). Some typical lines with in the file SYSDEF.PA are as follows:

```
CHEMAD, TEXT  ?DRAKE@?
        TEXT  ?MALSTROM@@?
        TEXT  ?SMITH@?
          .

          .
```

```
              .

        TEXT  ?HOLLINGS@@?
        TEXT  ?@@@@?
```

Translation Tables Format (First Level):

Starting at location LTRAN is a table which contains one entry for each column. This entry is the starting address of the translation table for this column. Typical entries are as follows:

```
*LTRAN
 CØØ      /Address of Col. Ø Table
 CØ1      /Address of Col. 1 Table
  .
  .
  .
 C39      /Address of Col. 39 Table
```

The entries of the translation table are coded with one of three formats depending on whether a null character string, a single character, or a character string is to result on translation. Typical entries are as follows:

```
CØ7, TEXT  ?AB?         /Convert A to B
     TEXT  ?C@?         /Convert C to Null String
     TEXT  ?DA@B@C@D?   /Convert D to ABC
     TEXT  ?EA?         /Convert E to A
     TEXT  ?@@?         /End of table
```

The first level translate tables are created by IDGEN.BA from the file TRNTBL.AR and placed within the file SYSDEF.PA.

Translation Tables Format (Second Level):

The second level of translation translate an input character string into an output character string. A table is present for each field on the card. Both the input and output character strings are 6 or less characters.

At location PUTI22 is a table containing one entry for each field. This entry is the starting address of a list which contains all possible character string inputs for this field. If this address is zero, then no table exists and the string for this field is not translated. Typical entries are as follows:

```
*PUTI22
 PPØ      / Address of Field Ø List
 PP1      / Address of Field 1 List
  .
  .
  .
 PP1Ø     / Address of Field 1Ø List
```

Each entry in these lists are represented by six 6-bit ASCII characters which are packed into three consecutive words. An entry containing all zeroes terminates the list. Typical entries in this input list are as follows:

```
PP3, TEXT  ?1@@@@@?
       .

       .

     TEXT  ?53@@@@?
     TEXT  ?@@@@@@?
```

The tables for the output string start at location PUTI21 with one entry for each field. This entry is the starting address of the list containing the output character strings for this field. These character strings have the same format as the first level translation tables in which one character is converted into a variable length character string. The relative location within the input list for which a match was found defines a 6-bit positive integer. This 6-bit integer is interpreted by the output tables as a 6-bit ASCII character. Typical entries are as follows:

```
        *PUTI21
         PØ                    /Field 0 List Address
          .
          .
         P1Ø                   /Field 1Ø List Address
    P3,  TEXT  ?AP@H?
         TEXT  ?BC@O@N@D?
          .
          .
         TEXT  ?@@?
```

Conversion Tables:

Starting at location CVRTBL+2, there are two consecutive locations allocated per parameter ID. If a given parameter has an internal ID code of K, then location CVRTBL+2*K contains the starting address of a parameter reduction block. The following location contains the address of a Methods ID list for this parameter. Typical entries are as follows:

```
        *CVRTBL
          Ø
          Ø
         PARXØ1    /Address of Parameter Reduction Block
          Ø
         PARXØ2
          Ø
          .
          .
```

```
PARXnn
 Ø
 .
 .
 .
```

The parameter reduction identification block contains 4 consecutive locations per method. The symbolic starting address of this block is a 6 character name consisting of PARXnn where nn is the 2 digit internal parameter ID code. The absolute value of the first entry identifies the number of constants required by the reduction algorithm with a negative value indicating that these constants are known to the algorithm. The second location is the starting address of the function subroutine to perform the indicated data reduction and is symbolically FXXmm. The third entry is the address of a buffer for this parameter which is symbolically BFnnkk where nn is the internal parameter ID code and kk is the internal methods ID code. The last entry indicates the number of function coordinate pairs which can be stored within this buffer. Typical entries are as follows:

```
PARXØ1,  3         /3 Constants
         FXXØ5     /Function 3
         BFØ1ØØ    /Buffer Address
         Ø         /No tables
PARXØ2,  Ø         /No Constants
         FXXØ4     /Function 4
         BFØ2ØØ    /Buffer Address
         2         /Two Coordinate Pairs
```

The first entry in the buffer at location BFnnkk is a positive integer identifying the number of measurement values expected. If constants are required by the reduction algorithm, these constants are next and are stored in floating-point format with 3 locations per floating-point variable. The remainder of the buffer is used for X-Y coordinate tables for the piecewise linear functions used by the redundancy reduction algorithm. The first entry in this X-Y coordinate portion of buffer is a positive integer which gives the number of X-Y coordinates present. Each X-Y coordinate consists of two floating-point numbers.

The program IDGEN.BA creates the file CVRTBL.AR which contains these tables. This file must be assembled with the files PARCRH.AR and FNCTN.AR to yield these tables.

Format Table:

The format table starts at location FORMAT. The information within this table is dynamically generated at execution time. When the measurement format checking option is in effect, these tables contain the expected format for each measurement. Each entry contains two 6-bit integers with one integer being field width while the second integer defines the number of placed to the right of the decimal.

Command Tables:

The command tables are found within CRH.PA. For each command, there is an entry in three different lists. These lists contain the command ID tags, the starting address of the command routines, and the ID tags expected for each command when the NTRAN option is enabled.

Field 3 Buffers:

Field 3 is used as temporary storage for the various routines. The starting addresses and the length of these buffers are determined by the program IDGEN.BA.

PAGE ZERO CONSTANTS:

The following tables define the usage of the page zero variables:

Pointers to Messages

ENDMSA	End
COLMSA	Col Count Exceeded
ERRMSA	Error
PFMSA	Pick Fail
HPRMSA	Hoppers?
NRMSA	Not Ready
RDMSA	Ready

Pointers to Tables

OUTBFA	Output Buffer
LTRANA	First Level Translation Table
ERBFA	Error Buffer
TMPBFA	Temporary Measurement Buffer
SPCA	Special Character Table
CVRTBL	Conversion Tables

Card Reader Input-Ring Buffer Printers

CFILL	Next Free Address in Buffer
CBUFA	Starting Address of Buffer
CBFEND	Negative of Ending Address of Buffer
CRCNT	Number of Characters in Buffer
CEMPTY	Address in Buffer of Next Character

Printer Ring Buffer Pointers

TPCNT	Number of Characters in Buffer
TPFLG	Nonzero when busy
TEMPTY	Address of Next Character to Print

TBUFA	Starting Address of Buffer
TFILL	Next Free Address
TBFEND	Negative of Ending Address

Card Reader Send Ring Buffer Pointers

CSCNT	Number of Characters in Buffer
CSBUFA	Starting Address
CSFILL	Next Free Address
CSFLG	Nonzero when Busy
CSEMP	Address of Next Character
CSBFEN	Negative of Ending Address

Variable Storage

CMDCOD	Command Code
WD1	Day and Point ID
WD2	Point and Period ID
WD3	Discrete and Comment ID
WD4	Quality ID
WD5	Chemist and Method ID
WD6	Accuracy ID
CONFLG	Continuation Code
LCHR	Last Character Input
CHR	Present Character Input
CRFLAG	Nonzero when CR received
ENDFLG	End Program when Nonzero
CHKFLG	Field Pointer

ERBF	Pointer to Error Buffer
ERFLG	Number of Errors
COL	Which column
CNTCOL	Count Columns

Remaining Entries

The remaining entries are either constants or subroutine entry points. Location 40-62 are dedicated to the floating point package.

MAIN PROGRAM:

The main program is located in Field Ø and starts at location 2ØØ. This program requires less than one page of storages. The remainder of INPUT.SV is either subroutines or tables used by this program.

FIELD Ø SUBROUTINES:

CRSD

If the Teletype is the input device, this subroutine causes a (#) to be printed to prompt the user for input. If the card reader is the input device, then the Control Y and Control Q are sent to the card reader to instruct the card reader to read the next card.

SEOF

This subroutine send a Carriage Return and Line Feed to the printer.

CRDY

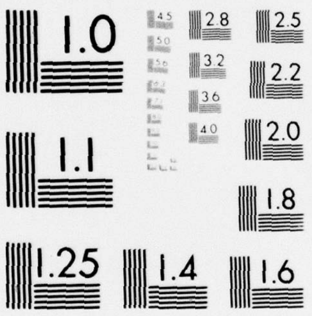
The error flag (ERFLAG) is cleared. If input is from the Teletype, then exit from this subroutine. For card input, this subroutine expects a control R from the card reader before returning. This subroutine will branch to END if an end condition is detected. The message CON is generated whenever an invalid character sequence is detected.

AD-A036 530 CLEMSON UNIV S C COLL OF ENGINEERING F/G 13/2
A SYSTEMS ANALYSIS OF WATER QUALITY SURVEY DESIGN. APPENDIX VII--ETC(U)
AUG 75 T L DRAKE DADA17-72-C-2152

UNCLASSIFIED NL

2 OF 2
ADA036530

END
DATE
FILMED
3-77



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

REPERR

When called, an error code appears in the left six bits of the AC. This subroutine adds the column number to this error code and stores this code in an error buffer.

CNTRL

When a control character is received on input from the card reader, this subroutine is called to decode this command sequence. The control codes are:

CNTRL	R	RETURN	Ready
CNTRL	I	RETURN	Pick Failed
CNTRL	G	RETURN	Hoppers
BACK	ARROW	RETURN	Not Ready
CNTRL	P	RETURN	End

The receipt of each control code will produce the appropriate message on the printer. If end condition is detected, the program branches to END.

TRSL

This subroutine is called with the character to be translated in the AC with the address of translation table following the subroutine call. If the Teletype is the input device, no translation occurs. For the card reader, an address pointer to the table entries corresponding to this character is returned provided the character is valid. The subroutine return address indicates whether the character is valid or not.

MSG

This subroutine prints a message on the printer. The subroutine is called with the starting address of message character string in the AC. This character string must be in Field CURØ.

DECON

This subroutine prints the positive binary integer in the AC as a two digit decimal number.

PLIST

If the input device is the Teletype, the character in the AC is passed to the subroutine PUTBF.

When the card reader is the input device, this subroutine transfers the character string pointed to by the subroutine TRSL to the subroutine PUTIT. The starting address of this character string is in the AC upon entry.

READ

Read in one character from the input device and make parity bit equal to 1. This subroutine checks to see if this character is found in a special character list. If special, the subroutine returns with the special character in the AC. For other characters, the character is converted to a 6-bit format and the return address is incremented by 1.

IERBF

The starting address of the error buffer is initialized.

ERROR

When errors are present, this routine prints on the printer the error messages coded in the error buffer. After ERROR is printed, each entry in the error buffer is converted and printed as an alphanumeric character following by two numeric characters.

Depending on the E option bit, the routine will pause until a *CNTRL R RETURN is received.*

REPER2

This subroutine reports errors from other fields. The subroutine REPERR doesn't have capability to determine field of calling program.

FERR

When errors are detected, control is often transferred by a JMP to this routine. This routine calls the subroutine ERROR to report all errors and starts the read of the next card.

FEND

Many functions when finished transfer control to this routine. If errors are present, control is transferred to FERR. Otherwise, the read of the next card is started.

PUTBH1

This is a table which specifies the address to PUTBF of the routine to process the fields on the card. The first entry is the address of processing routine for the command field, the second entry is the address of the processing routine for day field, etc.

PUTBH2

This is a table used by PUTBF which specifies the starting addresses of the various valid ID lists.

IDERR

Whenever an invalid ID or a null ID is detected, this subroutine is called. If invalid, the appropriate error code is entered into the error buffer. If null, the subroutine DEFID is called to determine whether a default ID is to used. When a new measurement is detected, the previous measurement is output.

FGETIN

This subroutine initializes the subroutine GETINT from any field.

FGETR

This subroutine gets a character by calling GET.

PUTBF

This subroutine receives the translated character strings. These character strings are segmented into fields by the presence of a space. The numeric field is segmented into measurements by the presence of certain spaces. The function performed by this subroutine is dependent on the field. The fields are converted to internal format by this subroutine.

When called, a 6-bit character is present in the AC. When the program is processing the numeric field, this character is tested to see if it is a leading space, excessive space, or trailing space as these spaces are ignored. The character is stored as an 8-bit character in the buffer at IDBUF and is printed on the printer.

When the subroutine is not in the numeric field, the character is tested to see if it is a space. If not a space, the character is printed and saved in the buffer at IDBUF. When a space, the subroutine branches to a routine to convert the character string in IDBUF to an internal format. The character string in IDBUF is first converted from 8-bit codes to 6-bit codes and packed in the floating-point AC. The resulting character string in the floating point AC is compared to the appropriate ID list for a match. The resulting ID code if valid is packed into either CMDCOD, WD1, WD2, WD3, WD4, WD5, or WD6. Otherwise, an error is reported.

<u>NULBF</u>

This subroutine fills the buffer starting at IDBUF in field CURI with null entries. The subroutines GET and PUT are initialized to unpack and pack this buffer with 8-bit ASCII characters.

<u>CLOSE</u>

This subroutine transfers the last output buffer with measurements to the output file and then closes this file.

<u>OUTFL</u>

The handler for the output file is first loaded and a file is opened for output on this output device. The device and file name are the first output file in the command decoder area. The output buffer is initialized to all zeroes.

<u>WROUT</u>

This subroutine transfers one block of measurements from a memory to the output device. During this transfer, the interrupt system is off.

<u>WAIT</u>

This subroutine places the program into a wait state until all I/O being performed via the interrupt system is finished. An empty printer buffer and the receipt of a RETURN from the input device signal completion.

<u>DPUT</u>

A measurement with ID tags in WD1, WD2, WD3, WD4, WD5, and WD6 and value in floating point AC is placed in the output buffer. If the buffer if filled by this measurement, this buffer is transferred to the output device. The output buffer is cleared after this transfer.

CHKID

A character string with 6 characters is compared with the entries of a list for a match. The starting address of this list in field CURO is found in the AC upon entry with the character string packed into the floating point AC. On return, the AC contains entry number where a match was found. A zero value indicates no match.

SAVEID

The internal ID in the AC is packed into either WD1, WD2, or WD3. When CHKFLG is zero, various locations within the program are initialized.

INTHDL

This is the interrupt handler. Characters are transferred between the I/O devices and the program's ring buffers on interrupts. After saving the machine status, the source of the interrupt is determined and with control transferred to the appropriate service routine.

The coding at TPSER transfers one character from the printer ring buffer to the printer. The coding at CRSSER transfers control characters from the (card reader output ring buffer to the card readers.)

The coding at CRRSER receives characters from either the keyboard or the card reader and places these characters in the input ring buffer. This presence of a CNTRL C will abort the program and transfer control to the monitor. The location CRGLAG is made nonzero whenever a RETURN is received.

BFRINT

This portion of the program contains three subroutines. These subroutines. These subroutines initialize the pointers utilized by the three ring buffers.

CSPUT

This subroutine places control characters in the card reader output ring buffer.

CRGET

This subroutine gets one character from the card reader input ring buffer.

TPPUT

This subroutine places characters in the printer ring buffer. This routine can be called from any field.

ERTRAN

This subroutine is called whenever a translation error is detected. This error is reported and a question mark is echoed on the printer.

DEVTT

This subroutine determines whether the input device is the Teletype or the card reader.

NRMCHK

The floating point package inputs one number and places this number in the floating point AC. During input, the format is determined. Whenever the format checking option is enable, this format is checked for validity.

The floating point package input routine has been modified to call the subroutine CHRFP to receive the next input character. This permits format checking and character strings in core to be used for input.

CNV86

Character strings, using 8-bit character formats, are converted to character strings which use a 6-bit character format. These character strings are 0 to 6 characters in length. The various lists of valid ID tags are represented internally by 6-bit characters while characters being input and translated are 8-bit.

This subroutine unpacks the 8-bit characters from the buffer IDBUF with the subroutine GET, checks to see if each character is a delimiter, converts the character to 6-bit, and packs the character in the floating point AC.

CHRFP

The floating package calls this subroutine to receive each character on input. This input character is unpacked from the IDBUF. The format of character string passed to the floating point package is determined.

COMFR

This subroutine is called by NRMCHK to compute the format of the character string passed the floating point package.

CHKFR

The format of the actual input is compared with the desired input format. When format error option is enable, a format error is generated whenever they differ.

FNCTX

Various data reduction algorithms use calibration curves which are piecewise linear functions. When called, the address of X and the address of a table containing the coordinate pairs, (X_1,Y_1), (X_2, Y_2), . . ., (X_n,Y_m) are supplied. Upon return, the function value is found in the floating point AC.

The function value is required to be bounded by both a minimum value and a maximum value whenever the parameter ID tags of this limiting values match the current measurement ID. The comment ID is changed whenever the minimum or maximum limit values are used.

FDISP

The main program branches to this routine whenever a command line has been received. The routine determines the starting address of a routine to process this command line and then branches to this command routine. The table in field 1 with a starting address of FDPTCH gives the addresses of these routines. Whenever continuation is enabled, these starting addresses are found in the table with a starting address of CONTN1. Continuation is used whenever more than one command line is required to define a function.

VALID

Whenever a measurement is received, this subroutine checks to see if a valid set of ID tags are present. The Day, Point, Period, Parameter, and Chemist ID tags must be nonzero. The Discrete ID must be zero for continuous measurements and nonzero for discrete measurements.

DEFID

This routine is called whenever an ID tag has not been specified by the user. A default ID will be used when specified provided a measurement is being entered.

Each default ID requires different coding. The table at DEF100 gives the starting address of the routine to use to generate this default ID.

PUTIT

Two levels of translations are provided by the input program. This subroutine provides the second level of translation.

Each character received from the card reader is translated into a variable length character string by the subroutines TRSL and PLIST. The subroutine PLIST calls PUTIT once for each character in this variable length string.

The subroutine PUTIT determines which field is being received by examining the CHKFLG. If this field doesn't require a second level of translation, this character is passed directly to PUTBF. If an additional level of translation is required, this character is packed in a buffer provided it is not a space. When a space is detedted, the subroutine PUTI70 is called to perform this translation and pass the translated character string to PUTBF.

The table in field CURØ with a starting address contains one entry for each ID field. A zero entry indicates that additional translation is not required. A nonzero entry is the address of the second translation table.

<u>SLASTV</u>

The last valid measurement is used to initialize a sum. This sum is used with the duplicate data option. The ID tags for this measurement are also saved.

<u>DMPLST</u>

Duplicate measurements are accepted by the program with the final measurement value being the average of these measurements. The sum of these measurements must be temporarily stored until the next measurement has been entered. If these measurements have different ID tags, then the average must be computed from this sum in temporary storage and then placed in the output buffer.

The subroutine DMPLST divides the accumulated sum by the number of measurements and places the result in the floating AC. The ID tags are restored to the value saved by the subroutine SLASTV. This measurement is then transferred to the output buffer with the average printed on the printer.

DUPDAT

This routine is branched to from DUPDA2. The measurement in the floating AC is added to the sum and the number of duplicate measurements is incremented by one. This coding is executed whenever duplicate measurements are entered.

DOCONV

This subroutine is called to perform the indicated data reduction of the character string which defines the numeric field. The flag CNVFLG is first checked to see if a conversion is indicated. If zero, this subroutine expects to find one measurement in the numeric field and converts this character string to a floating point result and places this result in the floating point AC.

If a conversion is indicated, this subroutine examines the information present in the conversion tables for this parameter. This subroutine expects to find the constants present for the data reduction algorithm. Next, the conversion tables are checked to see if a data reduction algorithm is specified. The character string which defines the numeric field is converted and placed in the TMPBFR buffer. The appropriate data reduction subroutine is then called with the result being returned in the floating point AC.

The subroutine prints the resulting measurement on the printer. This is accomplished by calling the subroutine DOCOUT.

PUTI7Ø

This subroutine is associated with the second level of translation. This subroutine is called when the end of a field has been detected. The character string which defines this field is packed into the floating point AC as six 6-bit characters. This character string is then compared to the input tables for the second level of translation to find a match.

Once a match is found, the translated character string is passed to PUTBF by the subroutine PUTI5Ø.

PUTI5Ø

This subroutine is called by PUTI7Ø to unpack the translated character string and pass each character to PUTBF.

NRINC

This subroutine converts the character string which defines the numeric field to floating point values and places the resulting values in a buffer specified by the calling program. The number of values expected is specified by the calling program with an error being generated if they do not agree in number.

If fewer measurements are found than expected, this subroutine expects these to be supplied on additional cards whenever this option has been selected. By default, the option to have data on multiple cards is disabled.

DOCOUT

This subroutine does the output for the DOCONV subroutine. The measurement to be printed is first checked for a limit violation by the subroutine CHRNG. The measurement or the measurement limit is then printed.

DEFIØØ

This is a default table which gives the starting address of a program for each field to provide a default ID for this field.

DEFIØ3

The default ID tags are packed into this table.

DEFI3Ø

This code is called by the DEFID subroutine to branch to the routine specified by the table at DEFIØØ.

SKPNID

This subroutine is called by the main program to skip designated fields in the command string and use the default ID for these skipped fields. Fields are skipped only when it is a measurement.

CLCMDR

This subroutine sets the option bits in the OS/8 command decoder area to zero.

CNV86

This subroutine converts 6-bit ASCII characters to 8-bit ASCII characters.

Field 2 Routines

FEND2

Branch to FEND in field Ø. The functions in field CUR4 branch to FEND2 when the function is complete.

FERR2

Branch to FERR in field Ø. The functions in this field branch to FERR2 whenever an error is detected.

PUTP2

This subroutine places one character in the printer ring buffer.

SCONFL

The continuation flag in field Ø is set to value of the AC by this subroutine.

RPER2

This field uses this subroutine to report errors.

SCRLF

A carriage return and line feed are placed in the printer ring buffer by this subroutine.

RPER3

Report error in this field and exit using error return.

PINTEG

Print on the printer the value of the integer in the AC.

NOID

This subroutine expects to find a null Day, Point, Period, Parameter, Method, Discrete, Quality, Accuracy, and Comment ID. An error results if this condition is not found.

NOVAL

This subroutine expects to find a null numeric field. An error results if one or more measurements are found.

CHCHEM

This subroutine determines the first two characters of the chemist name from the internal chemist ID.

CNV2DG

This subroutine converts a positive integer to a two digit decimal number. This decimal number is returned as two packed 6-bit characters in the AC. This subroutine is used to get two characters of a file name given by a positive integer less than 100.

DISDAT

This is the DISDAT function to set an option bit to receive discrete data only. The option bit is in OS/8 command decoder table.

CONDAT

This is the CONDAT function which sets an option bit to receive continuous data only.

CONERR

This is the CONERR function which sets an option bit to continue on an error condition.

NOTRAN

This is the NTRAN function which sets an option bit to accept input from the Teletype keyboard. All characters received are sent to PUTBF without translation.

TRANSL

This is the TRAN function which sets an option bit to accept input from the card reader. All characters received from the card reader pass through two levels of translation prior to being sent to PUTBF.

SCDXY

This is the CN1 function which enables the system to receive function defining coordinates.

SSXY

This subroutine gets the Parameter ID and sets up the starting address of a table.

LSXYAD

The starting address of table which holds function defining coordinates is determined. The size of the table is also computed.

LSXY

This is the LSFNC function which prints the function defining coordinates for named parameter.

LDXY

This function reads in a (X,Y) coordinate for the CN1 function. This function is called on a DA command following the CN1 command.

FLDXY

This function is called on a EDA command following CN1 command to end coordinate input. The coordinates entered are transferred from a temporary buffer to tables provided for this storage.

GETID

This subroutine expects to find a valid Parameter and Method ID. Except possible for the Chemist ID, the remaining ID tags are expected to be zero. An error results whenever the ID tags differ from the expected. Many commands expect to find only a Parameter and Method ID. This subroutine returns with the address of a table defining the reduction algorithm for this parameter.

CNVAL

This is the CN function which receives constants for the data reduction algorithm.

TRANSF

This subroutine transfers a variable number of floating point numbers from one buffer to another buffer.

END

This is the END function which terminates data entry.

ONCON

This is the ON function which enables the system to perform data conversions.

OFCON

This is the OF function which disables the system from performing data conversions.

ADATA

The functions ADATA and NOADA are implemented. The value of ADATFG determines which option the program is used.

MN

The function MN is implemented by this coding. The value for MINVAL and the parameter are received and transferred to the FNCTX routine.

MX

The function MX is implemented. The MAXVAL and parameter are received and transferred to the FNCTX routine.

CHCAL

The function CHCAL is implemented.

DUPDAZ

The function DUPDAT is implemented.

CHKFN

The function CHKFN is implemented which receives a value of X and a parameter ID. The value of this function is computed and printed on the printer.

MRDAT

The ADATA commands permits the numeric field on cammonds and measurements to be on one or more cards. When multiple cards are used, the DA card will cause system to branch to this routine to receive more numeric information.

MRDAT1

When an insufficient number of values are found in numeric field, the subroutine NRINC branches to MRDATI to enable the program to use multiple cards when the ADATA option has been enabled.

MXVAL

The MXVAL function receives an upper limit and parameter for use by the subroutine DOCONV in limiting the upper measurement value.

MNVAL

The MNVAL function receives a lower limit and parameter for use by the subroutine DOCONV in limiting the lower measurement value.

CHRNG

The subroutine is used by the DOCONV routine keep the measurement value bounded by a lower and upper limit. Whenever a measurement violation occurs the measurement is set equal to the limit and the Comment ID is changed to note this limit violation. Both limits have Parameter ID designations, the Parameter ID of the measurement must agree before limit checking results.

<u>LGFNC</u>

The functions LGFNC, LØFNC, and LIFNC use this conding to initialize the program to accept (X,Y) coordinate information. Several registers are initialized.

<u>CALDAT</u>

This function receives a (X,Y) coordinate for the LGFNC, LØFNC, and LIFNC command. The various sums and sums of squares are updated so that the least squares coefficients can be calculated. The original (X,Y) coordinate data is saved when possible in the buffer starting at TMBFR + 6 for a future print out.

<u>CCALDA</u>

The EDA command has been received to terminate the LGFNC, LØFNC, and LIFNC computation. The various least square coefficients are computed and placed in the appropriate tables. If enabled, the results are tabulated by calling the function PRFNC.

<u>PRFNC</u>

This original (X,Y) coordinate data along with the computed Y values are tabulated. This tabulation permits the user to visually compare the accuracy of the least square fit just performed.

<u>EPRFNC</u>

The EPRFNC command enables the routine PRFNC to tabulat-e the input and computed data after a least square curve fit.

<u>DPRFNC</u>

The DPRNC command disables the routine PRFNC from tabulating the data following a least square curve fit.

GTINT

This subroutine receives one or two decimal digits character string from the numeric field. This character string must be terminated by either a RETURN or a SPACE character. This subroutine is used to get extension and chemist code from the numeric field.

FORMT

The FORM command determines the format of the measurements within the numeric field and stores this format information in the format tables. The program is enabled to do format checking.

FORMON

The ONFORM command enables the system to perform format checking on the numeric field of measurement commands. A bit in the OS/8 command decoder area is set.

FORMOF

The OFFORM command disables the system from checking measurement formats of measurement commands. A bit in the OS/8 command decoder area is cleared.

GDEFID

The GDFID command places the ID information found in the command string and enables the system to supply default ID tags when missing.

ONDID

The ONDID command enables the system to use default ID tags when missing in a measurement command.

OFFDID

The OFFDID command disables the system from using default ID tags.

LSBFR

The LBVF command lists the measurements which reside within the output buffer. This command prints only the primary ID tags. This is accomplished by converting each internal ID tag to an external ID tag via the ID tables.

LSBFRG

This subroutine is called by LSBFR perform the actual ID conversion and printing.

FNAME

The FNAM command opens an output file provided a file is not now open. The first two characters of file name are D1. The second two characters are the first two names in the chemist name. The last two characters are the internal Parameter ID code. The extension is supplied by the command and must be two decimal digits.

Prior to entering this file, the handler is loaded for the device DATA. The directory is searched for a file having either the same name or a file having the same name but with the first letter changed to A. If such a file, this is considered an error.

The name of the file with device is placed in the OS/8 command decoder area. The subroutines which open and close files are found in field Ø.

RFNAM

The command RFNAM does the same as FNAM except that it permits a file to be entered even though a file exists with the same name.

CNAME

The command CFILE does the same as FNAM except the name starts with D2.

DNAME

The command DFILE does the same as FNAM except the name starts with D3.

CMDCR

The command CMDCR loads the OS/8 command decoder to receive the name of the output file. Options may be specified in this command string.

SKPID

The command SKPID determines which ID tags are present in the command string. Whenever an ID is present, a corresponding bit in location DEVTT5 is set.

CHGTRN

The CHGTRN command receives two positive integers from the numeric field. These integers specify two columns in the first level translation table. The translation table for the first column is changed to the translation table for the second column.

NACHEM

The NACHEM command receives a two digit code for the chemist. This code is the internal ID code for this chemist. This code is placed in WD5 and the name of the chemist is printed.

PRCHEM

The PRCHEM command prints the name of the current chemist.